

IsoNs – Next Step G Code User Manual

www.promax.it



PROMAX

Motion
&
Control

The contained information in this handbook are only informative and they can being change without warning and they must not being understandings with some engagement from Promax srl. Promax srl does not assume responsibility or obligates for errors or inaccuracies that can be found in this handbook. Except how much granted from the license, no part of this publication can be reproduced, saved in a recording system or transmitted in whatever form or with any means, electronic, mechanical or recording system or otherwise without Promax srl authorization. Any reference to names of society or products have only demonstrative scope and it does not allude to some real organization.

Rev. 2.0.9 © Promax srl

1 INTRODUCTION

This manual explains ISO programming and all relative functions. The method of programming can change with the used PC model (keyboard, touch-screen), however main concept remains unchanged.

To manage ISO programs refer to USER INTERFACE manual.

2 ISONS RULES

2.1 BLOCK

A block is formed by one or more ISO function ended with CR/LF character.

The sequence of more blocks composes a program (defined PartProgram) that it is managed by user interface.

All characters forming the block must be UPPER-CASE.

IsoNs editor makes an automatic conversion of lower-case characters.

G1X100Y100Z10

2.2 LINE NUMBER OR BLOCK NUMBER

Line number or block number defines block position inside the PROGRAM.

Such position is useful in some features (block restart, retrace, etc.).

IsoNs ignores line marks Nxx before block, therefore the real line number is the number showed on ISO editor window.

ISO CODE	
1	N10 G1X100Y100
2	N30 G4F2

IsoNs ignores **N10** and **N30** characters, therefore the real line is showed on the left bar. 1 and 2 in this example.

2.3 PROGRAM RUNNING

Program running is made by IsoNs user interface. When in run, program can be PAUSED or STOPPED.

2.4 AXIS NAME

IsoNs don't use any particular convention to define axis type (linear or rotative) by name (X,Y, etc.). Definition must be setted only in configuration. IsoNs manages up to 9 axis with the following names:

X,Y,Z,A,B,C,U,V,W

2.5 HOMING

All axis refer to an HOME POSITION made after switch on. Home position follows an HOMING procedure selected by relative machine parameters.

2.6 WORK ORIGIN

IsoNs manages up to 256 different WORK ORIGINS. Each origin defines a reference point of zero, all following positions refers to this point. Origins can be set both manually by interface and with dedicated instructions inserted in PartProgram. Furthermore they can be SUSPENDED, RESTORED or DISABLED.

2.7 WORK OFFSET

As well origin, IsoNs can manage up to 256 WORK OFFSET. An OFFSET works as a new origin manageable only with PartProgram. Also OFFSETS can be SUSPENDED, RESTORED or DISABLED.

2.8 HEADS

IsoNs manages 256 working HEADS for a single machine.

Each head refers to machine origin and it is automatically pre-setted (do not need to use WORK ORIGIN e WORK OFFSET).

Used head can be recall by PartProgram with Hn instruction. It programs automatically head offset of all axis (referred to machine origin).

2.9 MODAL FUNCTIONS

MODAL functions remains active for all current PartProgram and also the subsequent ones until a (or more) particular function disables it. G0,G1,G2,G3 etc. are modal functions and therefore it doesn't need repeat them until another function exclude it. G1 excludes G0,G2 e G3, G2 excludes G0,G1 e G3 etc.

```
G1X100Y100
X300Y120Z10
X450
```

In this example it is used modal G1 function. In block 2 and 3 it isn't necessary to insert G1 because it remains active.

2.10 RECOGNIZED CODES

Almost all recognized codes are highlighted with a colour. However, in the few cases when it doesn't happen, it doesn't mean instruction isn't recognized.

When IsoNs editor doesn't recognize an instruction it is underlined with RED and it shows error window.

2.11 NUMERIC VALUES SETTING

IsoNs works with two numeric values types:

INTEGER FLOATING

Integer values are used for all defined quantities (Loop numbers, etc.). When it is insert a floating value instead of integer one, IsoNs generate an error warning the user.

Floating values are used for all REAL quantities (axis position, radius, etc.).
IsoNs uses as decimal separator character "." (DOT).

2.12 PROGRAM REMARKS

IsoNs don't use CNC classic convention for program remarks.

Remarks starts with "//" characters and terminates with end line. IsoNs editor highlights it with GREEN colour.

Remarks can be insert also on the right of block.

Remarks are ignored by PartProgram and all its content are discarded.

```
// This is a remark
G1X100Y100 // This is a remark too
```

3 ISO NS INSTRUCTIONS

3.1 RECOGNIZED G CODES

Gcode	Description
<u>G0</u>	RAPID positioning
<u>G1</u>	Linear Interpolation at F programmed speed
<u>G2</u>	CW circular interpolation
<u>G3</u>	CCW circular interpolation
<u>G4</u>	Timed pause
<u>G10</u>	Enable external axis OVERRIDE ASSI on potentiometer
<u>G11</u>	Disable external axis OVERRIDE ASSI on potentiometer
<u>G17</u>	Set work plane on X-Y
<u>G18</u>	Set work plane on Z-X
<u>G19</u>	Set work plane on Y-Z
<u>G20</u>	Axes in Inch
<u>G21</u>	Axes in millimeters
<u>G22</u>	Change axis of the work plane
<u>G23</u>	Restore axis of the work plane (disable G22)
<u>G24</u>	Enable horizontal mirror
<u>G25</u>	Disable horizontal mirror G24
<u>G26</u>	Change a couple of axis
<u>G27</u>	Suspend G26
<u>G28</u>	Restore G26
<u>G30</u>	Enable automatic insert of fillet on edges
<u>G31</u>	Suspend G30
<u>G32</u>	Restore G30
<u>G33</u>	Enable automatic insert of bevel on edges
<u>G34</u>	Suspend G33
<u>G35</u>	Restore G33
<u>G36</u>	Gestione assi rotativi
<u>G40</u>	Disable TOOL RADIUS compensation
<u>G41</u>	Enable TOOL RADIUS compensation LEFT
<u>G42</u>	Enable TOOL RADIUS compensation RIGHT
<u>G43</u>	Enable TOOL LENGTH compensation
<u>G44</u>	Disable TOOL LENGTH compensation
<u>G45</u>	Enable TOOL LENGTH compensation, parameter from TOOL TABLE
<u>G46</u>	Disable G45
<u>G47</u>	Set mode of start/stop TOOL compensation G41 G42

<u>G48</u>	Define Depth Axis
<u>G49</u>	MILD MODE – Edge Smoothing
<u>G50</u>	Working plane rotation
<u>G51</u>	Suspend plane rotation G50
<u>G52</u>	Restore plane rotation G50
<u>G53</u>	As G98 (for compatibility to other cn)
<u>G54</u> <u>G54.n</u>	Work origin from memory, AXIS X (or axis selected by following parameters XYZ etc
<u>G55</u>	Work origin from memory, AXIS Y
<u>G56</u>	Work origin from memory, AXIS Z
<u>G57</u>	Work origin from memory, AXIS XY
<u>G58</u>	Work origin from memory, AXIS YZ
<u>G59</u>	Work origin from memory, AXIS XZ
<u>G60</u>	Enable FAST interpolation without stop on segments
<u>G61</u>	Enable interpolation with stop on segments
<u>G62</u>	Wait stop axis
<u>G63</u>	G1 Enable PX_MOVETO (3D interpolation outside selected plane) without stop
<u>G64</u>	G1 Enable interpolation on selected plane PX_LINETO (2D plane interpolation
<u>G65</u>	Enable 3D interpolation PX_MOVETO with calculated stop on edge by parameters
<u>G66</u>	AFC – Adaptive Feed Control
<u>G67</u>	PX_MOVETO for movement outside plane and PX_LINETO for inside ones
<u>G68</u>	Always using of PX_LINETO in G1 “TRANSPORTED AXIS”
<u>G69</u>	LHK – Depth of buffer Look Ahead on virtual CN
<u>G70</u>	Set work plane on a generic couple of axis
<u>G71</u>	Start axis home searching
<u>G72</u>	Enable N.U.R.B.S (Not Uniform Rational BSpline) filter
<u>G73</u>	Enable Noise filter
<u>G74</u>	Enable RLS (Remove Len Segment) filter
<u>G75</u>	Use G64 for movement inside work plane and G65 for movement outside work plane
<u>G80</u>	Forced pause by code
<u>G81</u>	Management secondary axes LIMITS
<u>G82</u>	Work ORIGIN at current position with sensor offset
<u>G83</u>	Update CPU1 counter
<u>G84</u>	Preset CPU axis
<u>G85</u>	Work OFFSET at current position with sensor offset
<u>G86</u>	Hardware preset axis on module 360 degrees
<u>G87</u>	Suspend head offset selected with Hn
<u>G88</u>	Restore head offset selected with Hn

<u>G89</u>	HardWare Preset of axis position
<u>G90</u>	Positioning/intepolating with ABSOLUTE position
<u>G91</u>	Positioning/intepolating with INCREMENTAL position
<u>G92</u>	Work ORIGIN at current position
<u>G93</u>	Work OFFSET at selected position
<u>G94</u>	Work ORIGIN at selected position
<u>G95</u>	Work OFFSET at current position
<u>G96</u>	Suspend work offset G93-G95
<u>G97</u>	Restore work offset G92-G94
<u>G98</u>	Suspend work origin G92-G94
<u>G99</u>	Restore work origin G92-G94
<u>G100</u>	Synchronous comand for VIRTUAL axis
<u>G101</u>	AXIS STOP comand
<u>G102</u>	Start sensor acquisition
<u>G103</u>	Set RTCP parameters
<u>G104</u>	Enable RTCP
<u>G105</u>	Suspend RTCP
<u>G106</u>	Smoothing Filter
<u>G120</u>	Enable vertical mirror
<u>G121</u>	Disable vertical mirror G20
<u>G940</u>	MOVE axes excluding WORK ORIGIN only in the actual block
<u>G1050</u>	Disable Scaling
<u>G1051</u>	ENABLE Scaling

4 OTHERS RECOGNIZED CODES

Gcode	Description
<u>X</u>	Axis X position
<u>Y</u>	Axis Y position
<u>Z</u>	Axis Z position
<u>A</u>	Axis A position
<u>B</u>	Axis B position
<u>C</u>	Axis C position
<u>U</u>	Axis U position
<u>V</u>	Axis V position
<u>W</u>	Axis W position
<u>DX</u>	Axis X relative position
<u>DY</u>	Axis Y relative position
<u>DZ</u>	Axis Z relative position

<u>DA</u>	Axis A relative position
<u>DB</u>	Axis B relative position
<u>DC</u>	Axis C relative position
<u>DU</u>	Axis U relative position
<u>DV</u>	Axis W relative position
<u>DW</u>	Axis W relative position
<u>OX</u>	Set Origin number for axis X
<u>OY</u>	Set Origin number for axis Y
<u>OZ</u>	Set Origin number for axis Z
<u>OA</u>	Set Origin number for axis A
<u>OB</u>	Set Origin number for axis B
<u>OC</u>	Set Origin number for axis C
<u>OU</u>	Set Origin number for axis U
<u>OV</u>	Set Origin number for axis V
<u>OW</u>	Set Origin number for axis W
<u>DOX</u>	Origin Offset fo axis X
<u>DOY</u>	Origin Offset fo axis Y
<u>DOZ</u>	Origin Offset fo axis Z
<u>DOA</u>	Origin Offset fo axis A
<u>DOB</u>	Origin Offset fo axis B
<u>DOC</u>	Origin Offset fo axis C
<u>DOU</u>	Origin Offset fo axis U
<u>DOV</u>	Origin Offset fo axis V
<u>DOW</u>	Origin Offset fo axis W
<u>F</u>	Axis FEED
<u>S</u>	Spinale speed
N	Line number (optional)
<u>D</u>	Tool diameter
<u>M</u>	M function
<u>HM</u>	HM function
<u>H</u>	Select head
<u>T</u>	Select tool
<u>I</u>	First CENTER coordinate for G2 G3
<u>J</u>	Second CENTER coordinate for G2 G3
<u>R</u>	Radius for G2 G3
<u>USER_ZERO</u>	Index of WORK ORIGIN LIST
<u>USER_OFFSET</u>	Index of WORK OFFSET LIST

<u>PAUSE_MODE</u>	Defines the Pause function mode
-------------------	---------------------------------

4.1 PROGRAM FLOW INSTRUCTIONS

Gcode	Description
<u>IF</u>	Start IF cycle
<u>ELSE</u>	Relative to IF cycle
<u>END IF</u>	End IF cycle
<u>LOOP</u>	Start LOOP cycle
<u>END LOOP</u>	End LOOP cycle
<u>GOTO</u>	Jump to LABEL or line number
<u>GOSUB</u>	Call of Soubrutine
<u>RETURN</u>	Retrun from Soubrutine
<u>@</u>	LABEL definition
<u>//</u>	Start remarks
<u>END_PROGRAM</u>	End of program
<u>WAIT_INPUT</u>	Wait of digital input with Time Out
<u>ERROR</u>	Force stop program with error

4.2 GENERIC INSTRUCTIONS

Gcode	Description
<u>SDO_DL</u>	Sdo download (can open)
<u>SDO_UL</u>	Sdo upload (can open)
<u>GET</u>	Read parameter of HM function
<u>READ_PARMAC</u>	Read machine parameter
<u>WRITE_PARMAC</u>	Write machine parameter
<u>LOAD_VAR</u>	Load a file of variable
<u>GET_VAR</u>	Read a variable from loaded file
<u>WRITE_VAR</u>	Write a variable in memory list
<u>SAVE_VAR</u>	Save a file of variable on harddisk
<u>DIM_VAR</u>	Set dimension of memory list
<u>FILE_EXISTS</u>	Check if file exists
<u>REMOVE_VAR</u>	Remove a value from current list
<u>CLEAR_VAR</u>	Remove ALL values from current list
<u>OPT</u>	Compiler option
<u>IMPORT</u>	Import external file
<u>END_IMPORT</u>	End of Import program
<u>PA(n,par)</u>	Set ABSOLUTE target positioner
<u>PD(n,par)</u>	Set REALTIME target positioner
<u>PF(n)</u>	Set FEED positioner
<u>PS(n)</u>	STOP positioner
<u>PM(n,par)</u>	STATUS positioner
<u>RESUME T</u>	Resume the PartProgram from last instruction Tn used
<u>DIM</u>	ARRAY management
<u>SAVE T</u>	Save the tools table in the IsoNs cfg

4.3 MATH AND LOGICAL OPERATORS

Gcode	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
(	Open bracket
)	Cosed bracket
[	Start expresion for axis count ex: G1X[\$var+\$var1]
]	End expresion for axis count ex: G1X[\$var+\$var1]
^	POW
%	XOR
>	Greater
<	Less
>=	Greater or equal
<=	Less or equal
<>	Not equal
=	Equal
	Logic Or
&&	Logic And
	Or Bit
&	And Bit
!	Negate
~	Not Bit
>>	Shift bit right
<<	Shift bit left

4.4 MATH FUNCTIONS

Gcode	Description
<u>SIN</u>	Sinus
<u>COS</u>	Cosinus
<u>LOG</u>	Logarithm
<u>EXP</u>	Exponetial
<u>SQR</u>	Square root
<u>TAN</u>	Tangent
<u>ATAN</u>	Arctangent
<u>ASIN</u>	Arcsinus
<u>ACOS</u>	Arccos
<u>INT</u>	Integer part of float rounded
<u>FIX</u>	Integer part of float whitout rounded
<u>ABS</u>	Absolute value
<u>DRG</u>	Set degrees for COS,SIN,TAN,ACOS,ASIN,ATAN
<u>RAD</u>	Set radiants for COS,SIN,TAN,ACOS,ASIN,ATAN

4.5 VARIABLES and CONSTANTS

Gcode	Description
<u>NUMBER</u>	Numeric constant ex: 432.12
<u>\$VARNAME</u>	Generic double value ex: \$VAR1
<u>\$[Qn]</u>	Demand axis position ex: \$[Q0] - \$[Q100]
<u>\$[Rn]</u>	Actual axis position ex: \$[R0] - \$[R100]
<u>\$[In]</u>	Digital input ex: \$[I10]
<u>\$[On]</u>	Digital output ex: \$[O10]
<u>\$[Tn]</u>	TIMER ex: \$[T3]
<u>\$[Un]</u>	Tool parameter ex: \$[U12]
<u>\$[Cn]</u>	Axis counter ex: \$[C0]
<u>\$[Hn]</u>	Head parameter ex: \$[H1]
<u>\$[Xn]</u>	Special parameters Variables ex: \$[X1]
<u>\$[Yn]</u>	Work origin position ex: \$[Y0]
<u>\$[Wn]</u>	Work offset position ex: \$[W0]
<u>\$[Kn]</u>	User generic on CN
<u>\$[En]</u>	Positive Software limit axis set
<u>\$[Sn]</u>	Negative Software limit axis set

4.6 PREDEFINED VARIABLES

Gcode	Description
<u>\$ PARM 1</u>	Parameter 1 for M function
<u>\$ PARM 2</u>	Parameter 2 for M function
<u>\$ PARM 3</u>	Parameter 3 for M function
<u>\$ PARM 4</u>	Parameter 4 for M function
<u>\$ PARM 5</u>	Parameter 5 for M function

4.7 NsForms INSTRUCTIONS

Gcode	Description
<u>LIB.MESSAGE</u>	Show Message Box
<u>LIB.SHOWFORM</u>	Show NsForm
<u>LIB.CLOSEFORM</u>	Close NsForm
<u>LIB.FORMPROP</u>	Set Property NsForm
<u>LIB.FORMTEXT</u>	Set Caption NsForm
<u>LIB.ADDLABEL</u>	Add a NsLabel to NsForm
<u>LIB.LABELPROP</u>	Set NsLabel Property
<u>LIB.LABELTEXT</u>	Set Text NsLabel
<u>LIB.LABELPRINT</u>	Print IsoNs Variable to NsLabel
<u>LIB.ADDBUTTON</u>	Add a NsButton to NsForm
<u>LIB.BUTTONPROP</u>	Set NsButtonProperty
<u>LIB.BUTTONTEXT</u>	Set Text NsButton
<u>LIB.BUTTONPRINT</u>	Print IsoNs Variable to NsButton
<u>LIB.ADDINPUT</u>	Add a NsInput Object to NsForm
<u>LIB.INPUTPROP</u>	Set NsInput Property
<u>LIB.INPUTSETVALUE</u>	Set NsInput Value from IsoNs Variable

5 PROGRAM FLOW

ISO-Ns allows programming with an extension of standard ISO code. The adding instruction to control the program flow are BASIC style increasing programming performance. Therefore it's possible to manage conditional cycles, iterative cycles, jumps to labels, etc.

5.1 IF-ELSE-END_IF

It allows conditional execution of an instructions block depending of the result of expression. IF cycles can be nested without limits.

Syntax

```
IF condition
    [instr. true]
ELSE
    [instr. false]
END_IF
```

condition Mandatory. Any numeric expression with a result of True or False.

Instr. true Instructions block executed if condition is TRUE

Istr. false Optional. Instructions block executed if condition is FALSE

END_IF End of IF ELSE cycle

Ex:

```
IF $VAR=$VAR2*15+($VAR4+18)
    $VAR1=10
    G1X10Y20
ELSE
    $VAR1=0
    G1X0Y0
END_IF
```

5.2 LOOP - END_LOOP

Iterative cycle of the program. Instructions inside LOOP END_LOOP cycle are executed a number of time depended of expression result. LOOP cycles can be nested without limits.

Syntax

```
LOOP variable
    [instructions]
END_LOOP
```

variable Mandatory. Any variable typed \$name or numeric value

Instructions Instructions block executed in each cycle

END_LOOP End of LOOP cycle

Ex:

```
G91
$VAR=20+($VAR1*5)
LOOP $VAR
    G1X1.3Y1.2
    IF $VAR1=20
        GOTO EXIT_LOOP      // FORCED EXIT FROM LOOP
    END_IF
END_LOOP
@EXIT_LOOP
```

5.3 GOTO

Unconditional jump to a label or line number identified by Nxx

Syntax

GOTO labelname

GOTO @linenumber

labelname Mandatory. A label of program defined with @labelname

@linenumber Line number identified by Nxx. **if LineNumber = -1 restart the Partprogram**

Ex:

```
IF $VAR1=0
    GOTO CYCLE_A
END_IF
IF $VAR1=1
    GOTO @100 // GOTO @-1 JUMP TO BEGIN
END_IF
// EXECUTE CYCLE A
@ CYCLE_A
G1X100Y100
.....
.....
// EXECUTE CYCLE B
N100G1X10Y10
```

5.4 GOSUB – RETURN

Jump to and return from a subroutine defined by a label.

Syntax

GOSUB labelname

GOSUB @linenumber

labelname Mandatory. A label of program defined with @labelname

@linenumber Line number identified by Nxx

it is mandatory that an instructions block which start with label, must to end with a RETURN

Ex:

```
IF $VAR1=0
    GOSUB CYCLE_A
END_IF
IF $VAR1=1
    GOSUB @100
END_IF
// EXECUTE CYCLE A
@ CYCLE_A
G1X100Y100
RETURN
// EXECUTE CYCLE B
N100G1X10Y10
RETURN
```

5.5 LABEL

A LABEL defines a jump point that can be use with GOTO e GOSUB. It isn't possible to have two or more LABEL with the same name, because it can cause a system conflict.

LABELs are exclusive for every PART PROGRAM, therefore ISONS defines as private the LABELs that will be found in M or HM functions, so they don't have any effect in the PART PROGRAM.

It's a good thing to have LABEL with relevant names.

Syntax

@labelname

@ Mandatory. Identifier of LABEL
labelname Mandatory. Unique name of the label
 All characters excluded math and logic operator are allowed

Ex:

```
@ CYCLE_A
G1X100Y100
.....
.....
@ CYCLE_B
G1X10Y10
```

5.6 END_PROGRAM

It forces an end of program. IsoNs inserts automatically this instruction at end of program list (end of file). However in some cases it can be necessary to end program in a particular point other than end of file.

Syntax

END_PROGRAM

Ex:

```
IF $VAR1=0
    END_PROGRAM // END PROGRAM IN PREMATURE MODE
END_IF
```

5.7 WAIT_INPUT

It waits a digital input for a programmed time. If time elapses without input is active it is possible stop the program reporting an ALARM.

Syntax

WAIT_INPUT input state time alarm

input Mandatory. Digital input number from 0 to 255
state Mandatory. Logic state of input to continue the program (0 = OFF state , 1 = ON state)
time Time in sec. (0.1 resolution). If time elapses it's possible to signal an alarm stopping the PartProgram
alarm **TIME=0 infinite wait**
 Generated alarm code (refer to configuration file)
 0 = No allarm generated

Ex:

```
WAIT_INPUT 2 1 5.3 12 // Wait input 2 at logic state ON for 5.3 sec. and //
signal alarm 12 if condition is not happen
```

5.8 ERROR

It cause an exit from PartProgram reporting an error. This instruction is useful to generate exceptions of PartProgram at events occurrences. Execution is stopped with an alarm.

Syntax

ERROR err_code

err_code Mandatory. Error code (refer to configuration file).

Ex:

```
IF $VAR1=0
    ERROR 10      // STOP PARTPROGRAM WITH ALLARM 10
END_IF
```

5.9 RESUME_T

Resumes the part program from the last statement used Tn.

It is used for tool management Damaged. In practice, when the measurement tool detects that the tool is damaged, through **RESUME_T** you can redo the work from last tool change made.

The parameter that follows **RESUME_T** is the tool alternative to that Damaged. This parameter is taken from the table tools.

Syntax

RESUME_T Tool_nr

Tool_nr Alternative Tool Number

5.10 SAVE_T

Saves the tools table in IsoNs cfg file.

With PartProgram is possible modify the tools table value. **SAVE_T** saves in permanent mode the modify.

Syntax

SAVE_T

```
T1 // SELECTED T1
${U0}=10 // WRITE TOOL DIAMETER INDEX TABLE 0
${U1}=100.3 // WRITE TOOL LENGHT INDEX TABLE 1
SAVE_T // SAVE THE TOOLS TABLE
```

6 GENERIC INSTRUCTIONS

6.1 SDO_DL

CanOpen SDO Down-Load function (refer to CanOpen documentation).
Write data in a CanOpen node.

Syntax

SDO_DL node index subindex len_data \$var

node	Mandatory. CanOpen node
index	Mandatory. Object index
subindex	Mandatory. Object sub-index
len_data	Mandatory. Length of datas inside \$Var (1 char – 2 int – 4 long)
\$Var	Mandatory. Variable containing data.

Ex:

```
// SEND TO NODE 2 AT INDEX 24596 AND SUBIND. 0
// 4 BYTE FROM VARIABLE $VAR1
SDO_DL 2 24596 0 4 $VAR1
```

6.2 SDO_UL

CanOpen SDO Up-Load function (refer to CanOpen documentation).
Read data in a CanOpen node.

Syntax

SDO_UL node index subindex len_data \$var

node	Mandatory. CanOpen node
index	Mandatory. Object index
subindex	Mandatory. Object sub-index
len_data	Mandatory. Length of datas inside \$Var (1 char – 2 int – 4 long)
\$Var	Mandatory. Variable where data will be write

Ex:

```
// READ FROM NODE 2 AT INDEX 24596 AND SUBIND. 0
// 4 BYTE WRITED IN VARAIBLE $VAR1
SDO_UL 2 24596 0 4 $VAR1
```

6.3 GET

Read parameter for HM function

ATTENTION: PARAMETERS ARE IN REVERSE ORDER COMPARED TO HOW THEY WERE WRITED

Ex:

```
HM 10 20 30 40
..
GET $PAR1 $PAR2 $PAR3 $PAR4 // READ PARAMETERS
then
$PAR1=40 $PAR2=30 $PAR3=20 $PAR4=10
```

6.4 READ_PARMAC

Read a machine parameter with name indication.

Syntax

READ_PARMAC "pname" \$var

pname Mandatory. Parameter name to read (ex: VJOG_X)
\$Var Mandatory. Variable where parameter value will be write

A runtime error will be generated if "pname" doesn't exist
ATTENTION: The name of parameter must be inside "..."

Ex:

READ_PARMAC "VJOG_X" \$VAR1

6.5 WRITE_PARMAC

Write a machine parameter with name indication.

Syntax

WRITE_PARMAC "pname" \$var

pname Mandatory. Parameter name to read (ex: VJOG_X)
\$Var Mandatory. Variable containing value to be write

A runtime error will be generated if "pname" doesn't exist
ATTENTION: The name of parameter must be inside "..."
ATTENTION: write of parameters can be cause system errors if execute inappropriately

Ex:

\$VAR1=100

WRITE_PARMAC "VJOG_X" \$VAR1

6.6 OPT

Set the option of compiler for M functions.

Change the function M for actual PartProgram

Syntax

OPT Options Value

Options Options:

MSTOP,MSTART,MEND,MGOBLOCK,MGORETRACE,MERROR,MPAUSE,MGOPAUSE

Value New value for M functions

EX:

OPT MSTOP 10 //M STOP = 10 for actual PartProgram

OPT MERROR -1 //Disable M error for actual PartProgram

6.7 PAUSE_MODE

Defines the way in which CNC must act PAUSE.

The modes are as follows:

0	Normal Mode(Pause Enabled,M Pause Enabled,M Resume Pause Enabled)
1	Pause Enabled, M Pause Enabled, <u>M Resume Pause Disabled</u>
2	Pause Enabled, <u>M pause Disabled</u> , M Resume Pause Enabled
3	Pause Enabled, <u>M pause Disabled</u> , <u>M Resume Pause Disabled</u>
4	<u>PAUSE DISABLED ON CNC</u>

Syntax

PAUSE_MODE val

Val Value or expression

Pause_Mode used to disable or limit the functionality of the PAUSE cycles during some special work: Ex: TOOL CHANGE etc..

6.8 IMPORT

Import an external file with various formats.

The predefined format is ISONS. If present **NsImportFile.dll** file, this defines the import format.

Syntax

IMPORT "filename.ext"

filename It contains the absolute or relative file path.
 If **FileName** initiates with \$APPPATH,, the file is searched in the folder ISONS
 Import generates the events StartImport and EndImport when the file terminated execution
 The MAIN part program resume execution when Import File is terminated.
 import multiple files can be nested .

Ex:

IMPORT "\$APPPATH\PROJECT\IMPORT\TESTIMPORT.ISO" // Import from IsoNs folder

IMPORT "C:\FILEIMPORT\TESTIMPORT.ISO" // import from absolute path

6.9 END_IMPORT

Terminate in anticipated mode a program called from function IMPORT and return to back program calling

Syntax

END_IMPORT

Ex:

IF \$VAR=0

END_IMPORT

END_IF

7 NSFORMS INSTRUCTIONS

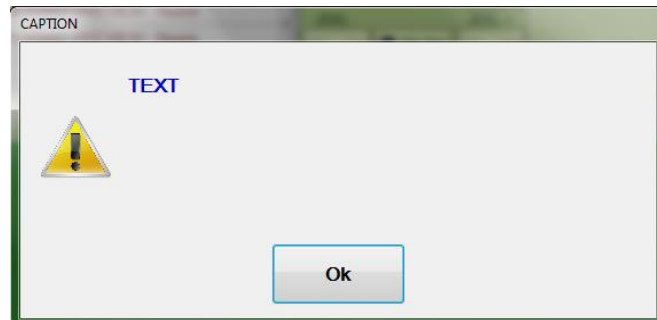
7.1 LIB.MESSAGE

Show a MessageBox with specified mode.

The program is suspended up to executed action in MessageBox

Syntax

LIB.MESSAGE "TEXT" "CAPTION" BUTTON ICON



TEXT

Body Message Text \N line feed carriage return

CAPTION

Message Title

BUTTON (value or variable)

- 0 OK
- 1 YES - NO

ICON (value or variable)

- 0 ERROR
- 1 WARNING
- 2 INFORMATION
- 3 QUESTION
- 4 NONE

Ex:

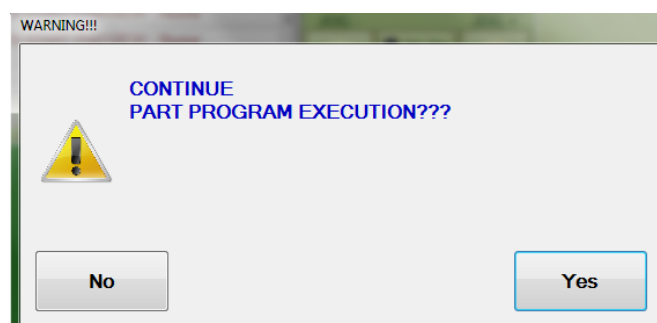
LIB.MESSAGE "CONTINUE\NPART PROGRAM EXECUTION???" "WARNING!!!" 1 1

\$BTN=\$[X13] // READ THE BUTTON PRESSED

IF \$BTN=2

END_PROGRAM // END PROGRAM IF PRESSED NO

END_IF



The **LIB.MESSAGE** instruction returned the code button pressed in the variable **\$[X13]**.
this contain

- 0 → Button **OK** pressed
- 1 → Button **SI** pressed
- 2 → Button **NO** pressed

7.2 LIB.SHOWFORM

Show a NsForm. The properties must be set by **LIB.FORMPROP** instruction.
The NsForm is in any case closed when the actual Part Program is finished.

Only one NsForm is displayed

Syntax

LIB.SHOWFORM

7.3 LIB.CLOSEFORM

Close a NsForm previously opened .

Syntax

LIB.CLOSEFORM

7.4 LIB.FORMPROP

Set the NsForm properties. The properties can be setted before a **LIB.SHOWFORM**

Syntax

LIB.FORMPROP "PROPNAME" PROPVALUE (value or variable)

PROPNAME (Property Name TEXT) :

VISIBLE

PropValue 0 (or < ZERO) NsForm Invisible

PropValue 1 (or > ZERO) NsForm Visible

WIDTH

PropValue > zero set **WIDTH** NSFORM

HEIGHT

PropValue > zero set **HEIGHT** NSFORM

LEFT

PropValue > zero Set the **LEFT** position

TOP

PropValue > zero Set the **TOP** position

BACKCLOR

PropValue Value 0 to 140.Set BACKCOLOR (see color table)

STARTPOSITION (must be set before SHOWFORM)

PropValue 0 (or < di ZERO) Use the **LEFT** e **TOP** position

PropValue 1 (or > di ZERO) NsForm is centered on screen

WINDOWSTATE

PropValue 0 (or < di ZERO) Use **WIDTH** e **HEIGHT** dimensions

PropValue 1 (or > di ZERO) Use a Max dimension

Ex:

```
LIB.FORMPROP "WIDTH" 500
LIB.FORMPROP "HEIGHT" 300
LIB.FORMPROP "BACKCOLOR" 113 //RED
LIB.FORMPROP "STARTPOSITION" 1 //CENTER SCREEN
LIB.FORMTEXT "CAPTION"
LIB.SHOWFORM
```



7.5 LIB.FORMTEXT

Set the NsForm Caption. The properties can be setted befor a LIB.SHOWFORM

Syntax

LIB.FORMTEXT "CAPTION"

CAPTION (TEXT) :

Text of Caption

Ex:

LIB.FORMTEXT "CAPTION"

7.6 LIB.ADDLABEL

Add a NsLabel to NsForm. The NsLabel properties can be setted befor a LIB.SHOWFORM

Syntax

LIB.ADDLABEL "NAME"

NAME (TEXT) :

Univocal Name of NsLabel Whitout special characters

Ex:

LIB.ADDLABEL "LBL"

7.7 LIB.LABELPROP

Set the NsLabel properties. The properties can be setted before a **LIB.SHOWFORM** but before, is necessary that the NsLabel is added to NsForm with **LIB.ADDLABEL**

Syntax

LIB.LABELPROP "LBLNAME" "PROPNAME" PROPVALUE (value or variable)

LBLNAME NsLabel Name

PROPNAME (Property Name TEXT) :

VISIBLE

PropValue 0 (< of ZERO) NsLabel Invisible

PropValue 1 (> of ZERO) NsLabel Visible (default)

AUTOSIZE

PropValue 0 (< of ZERO) NsLabel dimension by WHIDTH e HEIGHT

PropValue 1 (> of ZERO) NsLabel automatic dimension (default)

ENABLED

PropValue 0 (< of ZERO) NsLabel enabled (default)

PropValue 1 (> of ZERO) NsLabel disabled

WIDTH

PropValue > zero NsLabel Width (if autosize = 0)

HEIGHT

PropValue > zero NsLabel Height (if autosize = 0)

LEFT

PropValue > zero NsLabel Left position in NsForm

TOP

PropValue > zero NsLabel Top position in NsForm

DIMFONT

PropValue Value 6 to 100. Font Dimension

BACKCLOR

PropValue Value 0 to 140.Set BACKCOLOR (see color table)

FORECOLOR

PropValue Value 0 to 140.Set FORECOLOR (see color table)

STYLE (default 0)

PropValue:

0 Normal **1** *Italic* **2** **Bold** **3** **Bold Italic**

ALIGN (default 0)

PropValue:

0 Middle Left **1** Middle Center **2** Middle Right
3 Top Left **4** Top Center **5** Top Right
6 Bottom Left **7** Bottom Center **8** Bottom Right

BORDER (default 0)

PropValue:

0 None **1** *Fixed Single* **2** *Fixed 3D*

Ex:

```
LIB.FORMPROP "WIDTH" 500
LIB.FORMPROP "HEIGHT" 300
LIB.FORMPROP "STARTPOSITION" 1
LIB.FORMTEXT "TEST LABEL"
LIB.ADDLABEL "LBL"
LIB.LABELPROP "LBL" "AUTOSIZE" 0
LIB.LABELPROP "LBL" "WIDTH" 100
LIB.LABELPROP "LBL" "HEIGHT" 50
LIB.LABELPROP "LBL" "LEFT" 5
LIB.LABELPROP "LBL" "TOP" 10
LIB.LABELPROP "LBL" "DIMFONT" 12
LIB.LABELPROP "LBL" "ALIGN" 0
LIB.LABELPROP "LBL" "BACKCOLOR" 9
LIB.LABELPROP "LBL" "FORECOLOR" 137
LIB.LABELTEXT "LBL" "MY LABEL"
LIB.SHOWFORM
```

7.8 LIB.LABELTEXT

Write a Text in to NsLabel

Syntax

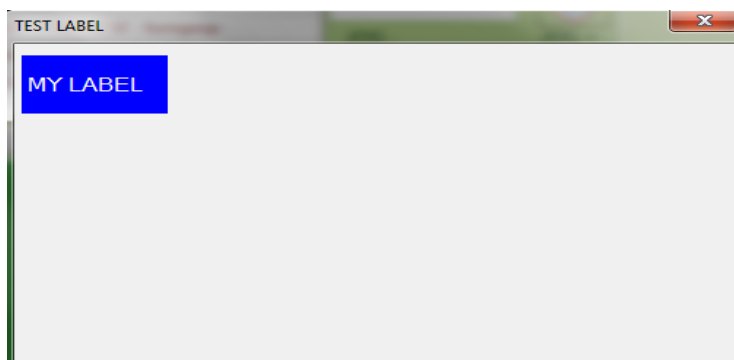
LIB.LABELTEXT "LBLNAME" "TEXT"

LBLNAME NsLabel Name

TEXT NsLabel Text

Ex:

```
LIB.LABELTEXT "LBL" " MY LABEL"
```



7.9 LIB.LABELPRINT

Write a IsoNs Variable in to NsLabel

Syntax

LIB.LABELPRINT "LBLNAME" "FORMAT" \$VAR

LBLNAME NsLabel Name

FORMAT Print Format:

"I" Only Integer value

"Fndec" Number of decimal

\$VAR IsoNs variable

Ex:

\$VAR=103.2569

LIB.LABELPRINT "LBL" "I" \$VAR //PRINT 103

LIB.LABELPRINT "LBL" "F1" \$VAR //PRINT 103.2

LIB.LABELPRINT "LBL" "F4" \$VAR // PRINT 103.2569

7.10 LIB.ADDBUTTON

Add a NsButton to NsForm. The NsButton properties can be setted befor a LIB.SHOWFORM

Syntax

LIB.ADDBUTTON "NAME"

NAME (TEXT) :

Univocal Name of NsButton Whitout special characters

Ex:

LIB.ADDBUTTON "BTN"

7.11 LIB.BUTTONPROP

Set the NsButton properties. The properties can be setted before a **LIB.SHOWFORM** but before, is necessary that the NsButton is added to NsForm with **LIB.ADDBUTTON**

Syntax

LIB.BUTTONPROP "BTNNAME" "PROPNAME" PROPVALUE (value or variable)

BTNNAME NsButton Name

PROPNAME (Property Name TEXT) :

VISIBLE

PropValue 0 (< of ZERO) NsButton Invisible

PropValue 1 (> of ZERO) NsButton Visible (default)

ENABLED

PropValue 0 (< of ZERO) NsButton enabled (default)

PropValue 1 (> of ZERO) NsButton disabled

WIDTH

PropValue > zero NsButton Width

HEIGHT

PropValue > zero NsButton Height

LEFT

PropValue > zero NsButton Left position in NsForm

TOP

PropValue > zero NsButton Top position in NsForm

DIMFONT

PropValue Value 6 to 100. Font Dimension

BACKCOLOR

PropValue Value 0 to 140.Set BACKCOLOR (see color table)

FORECOLOR

PropValue Value 0 to 140.Set FORECOLOR (see color table)

STYLE (default 0)

PropValue:

0 Normal **1** *Italic* **2** **Bold** **3** **Bold Italic**

ALIGN (default 1)

PropValue:

0 Middle Left **1** Middle Center **2** Middle Right
3 Top Left **4** Top Center **5** Top Right
6 Bottom Left **7** Bottom Center **8** Bottom Right

DEST

Destination variable writing. Indicates the destination variable where the value is written to Source

LIB.BUTTONPROP "BTN" "DEST" \$VAR

SOURCE

Source variable. Value to write to DEST

When you press the value of the source variable is written to DEST.

If fixed number can not contain decimal places

LIB.BUTTONPROP "BTN" "SOURCE" \$VAR1

LIB.BUTTONPROP "BTN" "SOURCE" 1230

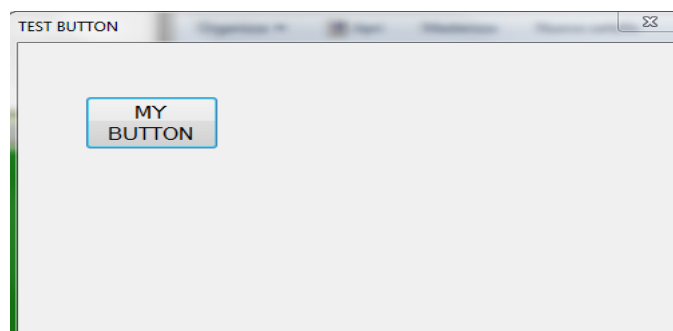
CLOSEFORM (default 0)

PropValue 0 (o < of ZERO) When press the NsForm is Not closed

PropValue 1 (o > of ZERO) When press the NsForm is closed

Ex:

```
LIB.FORMPROP "WIDTH" 500
LIB.FORMPROP "HEIGHT" 300
LIB.FORMPROP "STARTPOSITION" 1
LIB.FORMTEXT "TEST BUTTON"
LIB.ADDBUTTON "BTN"
LIB.BUTTONPROP "BTN" "WIDTH" 100
LIB.BUTTONPROP "BTN" "HEIGHT" 50
LIB.BUTTONPROP "BTN" "LEFT" 50
LIB.BUTTONPROP "BTN" "TOP" 50
LIB.BUTTONPROP "BTN" "DIMFONT" 12
LIB.BUTTONPROP "BTN" "ALIGN" 1
LIB.BUTTONTEXT "BTN" "MY BUTTON"
LIB.BUTTONPROP "BTN" "DEST" $VAR
$VAR=0
$VAR1=100.2
LIB.BUTTONPROP "BTN" "SOURCE" $VAR1
LIB.SHOWFORM
```



7.12 LIB.BUTTONTEXT

Write a Text in to NsButton

Syntax

LIB.BUTTONTEXT "BTNNAME" "TEXT"

LBLNAME NsButton Name

TEXT NsButton Text

Ex:

LIB.BUTTONTEXT "BTN" " MY BUTTON"

7.13 LIB.BUTTONPRINT

Write a IsoNs Variable in to NsButton

Syntax

LIB.BUTTONPRINT "BTNNAME" "FORMAT" \$VAR

BTNNAME NsButton Name

FORMAT Print Format:

"I" Only Integer value

"Fndec" Nymber of decimal

\$VAR IsoNs variable

Ex:

\$VAR=103.2569

LIB.BUTTONPRINT "BTN" "I" \$VAR //PRINT 103

LIB.BUTTONPRINT "BTN" "F1" \$VAR //PRINT 103.2

LIB.BUTTONPRINT "BTN" "F4" \$VAR // PRINT 103.2569

7.14 LIB.ADDINPUT

Add a NsInput to NsForm. The NsInput properties can be setted befor a LIB.SHOWFORM
The NsInput Object is used to write numerical value to IsoNs variable

Syntax

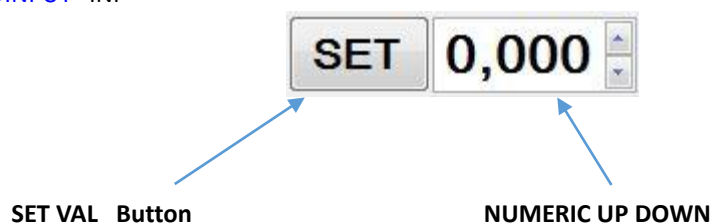
LIB.ADDINPUT "NAME"

NAME (TEXT) :

Univocal Name of NsInput Whitout special characters

Ex:

LIB.ADDINPUT "INP"



Button **SET** when pressed writes the value in IsoNs variable identified in "**DEST**"
If **AUTOSET=1** the value is automatically written when this is changed

7.15 LIB.INPUTPROP

Set the NsInput properties. The properties can be setted before a **LIB.SHOWFORM** but before, is necessary that the NsInput is added to NsForm with **LIB.ADDBUTTON**

Syntax

LIB.INPUTPROP "INPNAME" "PROPNAME" PROPVALUE (value or variable)

INPNAME NsInput Name

PROPNAME (Property Name TEXT) :

VISIBLE

PropValue 0 (< of ZERO) NsInput Invisible

PropValue 1 (> of ZERO) NsInput Visible (default)

ENABLED

PropValue 0 (< of ZERO) NsInput enabled (default)

PropValue 1 (> of ZERO) NsInput disabled

WIDTH

PropValue > zero NsInput Width

LEFT

PropValue > zero NsInput Left position in NsForm

TOP

PropValue > zero NsInput Top position in NsForm

DIMFONT

PropValue Value 6 to 100. Font Dimension

BACKCLOR

PropValue Value 0 to 140. Set BACKCOLOR (see color table)

FORECOLOR

PropValue Value 0 to 140. Set FORECOLOR (see color table)

ALIGN (default 0)

PropValue:

0 Left **1** Center **2** Right

DEST

Destination variable writing. Indicates the destination variable where the value is written

LIB.BUTTONPROP "BTN" "DEST" \$VAR

MIN

PropValue Minimum value accepted (double)

MAX

PropValue Maximum value accepted (double)

NDECIMAL

PropValue > zero Number of decimals displayed

INCREMENT

PropValue Increment value when pressed **up** - **down** buttons (double)

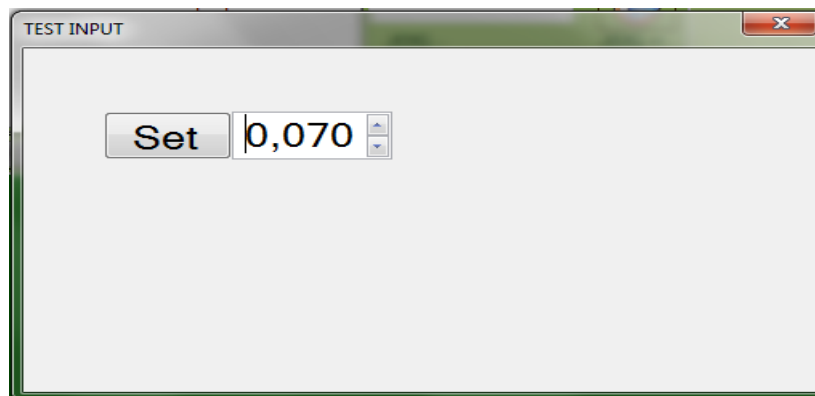
AUTOSET

PropValue 0 (< ZERO) The value is writes when pressed the button SET

PropValue 1 (> ZERO) The value is automatically writes when this changed

Es:

```
LIB.FORMPROP "WIDTH" 500
LIB.FORMPROP "HEIGHT" 300
LIB.FORMPROP "STARTPOSITION" 1
LIB.FORMTEXT "TEST INPUT"
LIB.ADDINPUT "INP"
LIB.INPUTPROP "INP" "WIDTH" 100
LIB.INPUTPROP "INP" "DIMFONT" 20
LIB.INPUTPROP "INP" "LEFT" 50
LIB.INPUTPROP "INP" "TOP" 50
LIB.INPUTPROP "INP" "ALIGN" 1
LIB.INPUTPROP "INP" "DEST" $VAR
LIB.INPUTPROP "INP" "MIN" 0
LIB.INPUTPROP "INP" "MAX" 100
LIB.INPUTPROP "INP" "NDECIMAL" 3
$VAR1=0.01
LIB.INPUTPROP "INP" "INCREMENT" $VAR1
$VAR=0
```



7.16 LIB.INPUTSETVALUE

Write a IsoNs Variable in to NsInput

Syntax

LIB.INPUTSETVALUE "INPNAME" \$VAR

INPNAME NsInput Name

\$VAR IsoNs variable

Ex:

\$VAR=103.2569

LIB.INPUTSETVALUE "INP" \$VAR //SET VALUE AT 103.2569

7.17 COLOR TABLE

Following color table to refer when the property is a color

0	ALICEBLUE		35	DARKSLATEGRAY		70	LIGHTSALMON	
1	ANTIQUEWHITE		36	DARKTURQUOISE		71	LIGHTSEAGREEN	
2	AQUA		37	DARKVIOLET		72	LIGHTSKYBLUE	
3	AQUAMARINE		38	DEEPPINK		73	LIGHTSLATEGRAY	
4	AZURE		39	DEEPSKYBLUE		74	LIGHTSTEELBLUE	
5	BEIGE		40	DIMGRAY		75	LIGHTYELLOW	
6	BISQUE		41	DODGERBLUE		76	LIME	
7	BLACK		42	FIREBRICK		77	LIMEGREEN	
8	BLANCHEDALMOND		43	FLORALWHITE		78	LINEN	
9	BLUE		44	FORESTGREEN		79	MAGENTA	
10	BLUEVIOLET		45	FUCHSIA		80	MAROON	
11	BROWN		46	GAINSBORO		81	MEDIUMAQUAMARINE	
12	BURLYWOOD		47	GHOSTWHITE		82	MEDIUMBLUE	
13	CADETBBLUE		48	GOLD		83	MEDIUMORCHID	
14	CHARTREUSE		49	GOLDENROD		84	MEDIUMPURPLE	
15	CHOCOLATE		50	GRAY		85	MEDIUMSEAGREEN	
16	CORAL		51	GREEN		86	MEDIUMSLATEBLUE	
17	CORNFLOWERBLUE		52	GREENYELLOW		87	MEDIUMSPRINGGREEN	
18	CORNSILK		53	HONEYDEW		88	MEDIUMTURQUOISE	
19	CRIMSON		54	HOTPINK		89	MEDIUMVIOLETRED	
20	CYAN		55	INDIANRED		90	MIDNIGHTBLUE	
21	DARKBLUE		56	INDIGO		91	MINTCREAM	
22	DARKCYAN		57	IVORY		92	MISTYROSE	
23	DARKGOLDENROD		58	KHAKI		93	MOCCASIN	
24	DARKGRAY		59	LAVENDER		94	NAVAJOWHITE	
25	DARKGREEN		60	LAVENDERBLUSH		95	NAVY	
26	DARKKHAKI		61	LAWNGREEN		96	OLDLACE	
27	DARKMAGENTA		62	LEMONCHIFFON		97	OLIVE	
28	DARKOLIVEGREEN		63	LIGHTBLUE		98	OLIVEDRAB	
29	DARKORANGE		64	LIGHTCORAL		99	ORANGE	
30	DARKORCHID		65	LIGHTCYAN		100	ORANGERED	
31	DARKRED		66	LIGHTGOLDENRODYELLOW		101	ORCHID	
32	DARKSALMON		67	LIGHTGRAY		102	PALEGOLDENROD	
33	DARKSEAGREEN		68	LIGHTGREEN		103	PALEGREEN	
34	DARKSLATEBLUE		69	LIGHTPINK		104	PALETURQUOISE	

105	PALEVIOLETRED		140	YELLOWGREEN	
106	PAPAYAWHIP				
107	PEACHPUFF				
108	PERU				
109	PINK				
110	PLUM				
111	POWDERBLUE				
112	PURPLE				
113	RED				
114	ROSYBROWN				
115	ROYALBLUE				
116	SADDLEBROWN				
117	SALMON				
118	SANDYBROWN				
119	SEAGREEN				
120	SEASHELL				
121	SIENNA				
122	SILVER				
123	SKYBLUE				
124	SLATEBLUE				
125	SLATEGRAY				
126	SNOW				
127	SPRINGGREEN				
128	STEELBLUE				
129	TAN				
130	TEAL				
131	THISTLE				
132	TOMATO				
133	TRANSPARENT				
134	TURQUOISE				
135	VIOLET				
136	WHEAT				
137	WHITE				
138	WHITESMOKE				
139	YELLOW				

8 MATH FUNCTIONS

8.1 SIN

Return the sinus of an angle in a DOUBLE type value.

Syntax

SIN (angle)

angle Mandatory. Angle in radiant (it can be an expression)

Ex:

`$VAR=SIN($VAR1*3)`

8.2 COS

Return the co-sinus of an angle in a DOUBLE type value.

Syntax

COS (angle)

angle Mandatory. Angle in radiant (it can be an expression)

Ex:

`$VAR=COS($VAR1*3)`

8.3 LOG

Return the base of natural logarithm in a DOUBLE type value.

Syntax

LOG (expression)

expression Mandatory. Expression.

Ex:

`$VAR=LOG($VAR1*3)`

8.4 EXP

Return the exponential of an expression in a DOUBLE type value.

Syntax

EXP (expression)

expression Mandatory. Expression

Ex:

`$VAR=EXP(10.15)`

8.5 INT

Return the integer part of a DOUBLE value rounded

Syntax

INT (expression)

expression Mandatory. Numeric expression

Es:

`$VAR=35.14`

`$VAR1=INT($VAR)`

8.6 FIX

Return the integer part of a DOUBLE value truncated

Syntax

FIX (expression)

expression Mandatory. Numeric expression

Es:

`$VAR=35.14`

`$VAR1=FIX($VAR)`

8.7 ABS

Return the absolute value

Syntax

ABS (expression)

expression Mandatory. Numeric expression

Es:

`$VAR=-35.14`

`$VAR1=ABS($VAR)`

8.8 DRG

Enable angle in **DEGREES** for COS,SIN,TAN,ACOS,ASIN,ATAN functions.

By default at each start of PART PROGRAM angle are setted in radiant

Syntax

DRG

8.9 RAD

Enable angle in **RADIANT** for COS,SIN,TAN,ACOS,ASIN,ATAN functions.

By default at each start of PART PROGRAM angle are setted in radiant

Syntax

RAD

8.10 SQR

Return the square root of an expression in DOUBLE type value.

Syntax

SQR (expression)

expression Mandatory. Expression

Ex:

`$VAR=SQR($VAR1)`

8.11 TAN

Return the tangent of an angle in a DOUBLE type value.

Syntax

TAN (angle)

angle Mandatory. Angle in radiant (it can be an expression)

Ex:

`$VAR=TAN($VAR1*3)`

8.12 ATAN

Return the arctangent of an expression. Result is from -3.14 to 3.14

Syntax

ATAN (expression)

expression Mandatory. Expression

Ex:

`$VAR=ATAN($VAR1*3)`

8.13 ASIN

Return the arc-sinus of an expression.

Syntax

ASIN (expression)

expression Mandatory. Expression

Ex:

`$VAR=ASIN($VAR1*3)`

8.14 ACOS

Return the arc-cos of an expression.

Syntax

ACOS (expression)

expression Mandatory. Expression

Ex:

\$VAR=ACOS(\$VAR1*3)

9 USING LOGIC OPERATORS

In this chapter will be explained how to use logical operator omitting classic math operators (+-*/).

9.1 ROUND BRACKETS ()

They separate level of expressions, doing priority to content inside the brackets.

9.2 ISO EXPRESSIONS and SQUARE BRACKETS []

The use of square brackets is not permitted in generic expression. They are valid only for specific type of variable and exclusively in the functions listed below. In these ISO instructions identify, in them, an expression or a variable. That is because in origin ISO code accepted only numeric fields, therefore to remain compatible at this standard but to have the possibility of a powerful and more flexible language, IsoNs introduced square brackets [] to identify an expression or variable.

ISO INSTRUCTIONS WHERE ARE PERMITTED SQUARE BRACKETS []

Axis positions	X,Y,Z,A,B,C,U,V,W - DX,DY,DZ,DA,DB,DC,DU,DV,DW
Axis Feed	F
Speed	S
Tool diameter	D
Arc Center	I,J
Arc Radius	R
Tool	T
Head	H
Origin index	USER_ZERO
Offset index	USER_OFFSET

Ex:

```
G1X1000           // STANDARD ISO GCODE
G1DX1000          // INCREMENTAL VALUE (THE SAME G91)
G1X[$VAR]         // USE OF VARIABLE AS AXIS POSITION
G1X[$VAR+$VAR1*SQR(18)] // USE OF EXPRESION AS AXIS POSITION
```

9.3 VARIABLE TEST

To test variable IsoNs uses classic conditional operators = > < <> >= <= !

They are inserted in **IF ELSE END_IF** cycles allowing to execute cycles conditioned to one or more variable. Equal operators (=) can be combined to other logical operators OR (|) AND (&&). Round brackets can be used to prioritize expressions.

Ex:

```
IF $VAR=0 && $VAR1=5
    G1X100
ELSE
    G1X0
END_IF
```

9.4 BIT CONTROL

IsoNs makes available also operator to control the state of a single bit of a variable. They can be used in **IF ELSE END_IF** cycles or to **set/reset** single bit of a variable. It is also possible make a shift of the bits (left or right) and negate its state. All bit operations can be made only on INTEGER variables (excluding therefore decimal part).

Ex:

```

IF $VAR & 4           // TEST BIT 3 OF $VAR
    G1X100 // BIT IS ON
ELSE
    G1X0           // BIT IS OFF
END_IF

$VAR=$VAR | 1 // SET BIT 1 OF $VAR
$VAR=$VAR & 1 // RESET ALL BITS EXCEPT BIT 1 OF $VAR

$VAR=1
$VAR=$VAR<<3 // $VAR IS SHIFTED LEFT 3 TIME
    
```

10 VARIABLE TYPES

IsoNs introduced VARIABLE to increase programming facilities. There are two types of variable: INTEGER o DOUBLE (floating point). Variables can be combined in expression as all other common languages.

10.1 NUMERIC CONSTANTS

All explicit numeric values are numeric constants. Decimal point can be used.

Ex:

```

G1X100.13
$VAR=25.143
    
```

10.2 GENERIC VARIABLES “\$”

All generic variable are DOUBLE type and they can be used to store values inside PartProgram. When they are inserted in ISO instructions (X,Y,Z ecc) it must be delimited between **square brackets [...]**.

Minimum value - **1.79769313486232 E308**
 Maximum value **1.79769313486232 E308**

The maximum number of available variables is 2048 and are recognized by \$ prefix followed by an ALPHANUMERIC NAME.

Ex:

```

$VAR=1
$VAR=$VAR1*5
$VAR=SQR($VAR1)
G1X[$VAR]
    
```

10.3 AXIS POSITION VARIABLES

These variables contain the position value of axis and accordingly they are read only.

The maximum number of these variables is 9. They can be used in PartProgram to read axis position in a given time.

Demand axis position - ReadOnly

\$(Qn) With **n** from **0** to **8** it identifies the axis. The value of **DEMAND POSITION** is written in the variable and it refers to a **RELATIVE** value inclusive of WORK ORIGIN and OFFSET

With **n** from **100** to **108** it identifies the axis (subtracting 100). . The value of **DEMAND POSITION** is written in the variable and it refers to an **ABSOLUTE** position from HOME position

Actual axis position - ReadOnly

\$(Rn) With **n** from **0** to **8** it identifies the axis. The value of **ACTUAL POSITION** is written in the variable and it refers to a **RELATIVE** value inclusive of WORK ORIGIN and OFFSET

With **n** from **100** to **108** it identifies the axis (subtracting 100). . The value of **ACTUAL POSITION** is written in the variable and it refers to an **ABSOLUTE** position from HOME position

Relative position Absolute position

n=0 AXIS X	n=100 AXIS X
n=1 AXIS Y	n=101 AXIS Y
n=2 AXIS Z	n=102 ASSE Z
...etc.	...etc.

Ex:

```
IF $(Q0)>100.3 // TEST OF X AXIS DEMAND POSITION
    G101
END_IF
$VAR=$(R1) // READ ACTUAL POSITION OF Y AXIS
```

10.4 DIGITAL INPUT/OUTPUT VARIABLES

These variables allow to read all digital input or set/reset all digital output of CN. Rember digital output can be shared with PLC cycle on CN. This facilities is useful to make a simple PLC structure directly from ISO language without CN programming. I/O variables consider only 0 and 1 values. ***These variables when inserted in PartProgram allow direct access to I/O without waiting axis movement sync. Therefore in specific cases can need to use G62 code (wait end movement) to have sync between axis and I/O.***

Input Variables – Read only

\$(In) Where **n** is a value from **0** to **255** and it identifies CN input
Value=0, Input OFF - Value=1, Input ON

Output Variables – Read/Write

\$(On) Where **n** is a value from **0** to **255** and it identifies CN output
Value=0, Input OFF - Value=1, Input ON

Ex:

```
IF $(I0)=1 // IF 1st INPUT IS ACTIVE
    $(O5)=1 // SET 6th OUTPUT
ELSE
    $(O5)=0 // ELSE CLEAR IT
END_IF
```

10.5 TIMER VARIABLES

These variables are useful to set system TIMERS. Timers can be used to make delays, time-outs, etc. The maximum numbers of timers available is 10. TIMERS are written with a time in milliseconds. After TIMER is set, it starts automatically to decrement until its value reach 0.

Timer Variables – Read/Write

\$[Tn] Where **n** is a value from **0** to **9** and it identifies the timer number

Ex:

```

$[T0]=1500          // SET TIMER 0 AT 1550 MILLISECONDS (1.5 SECONDS)
@CICLO
IF $[I0]=1
    GOTO EXIT      // EXIT IF INPUT 1 IS ON
END_IF
IF $[T0]=0          // TEST TIMER
    ERROR 10       // IF TIMER ELAPSES STOP PARTPROGRAM WITH ERROR 10
END_IF
GOTO CICLO          // REPEAT CYCLE
@EXIT

```

10.6 TOOL TABLE VARIABLES

They allow to read PARAMETERS from selected TOOL TABLE with **Tn**. The list below describes all parameters available for each TOOL TABLE and, as showed, they are up to maximum of 20 parameters per each tools.

Tool table variable – Read/Write

\$[Un] Where **n** is a value from **0** to **19** indicating the parameter address

Ex:

\$VAR=\$[U0] **// READ DIAMETER OF THE SELECTED TOOL**

Address	Description
0	Tool diameter
1	Tool length
2	Maximum tool rpm speed
3	User Define 1
4	User Define 2
5	User Define 3
6	User Define 4
7	User Define 5
8	User Define 6
9	User Define 7
10	User Define 8
11	User Define 9
12	User Define 10
13	User Define 11
14	User Define 12
15	User Define 13
16	User Define 14
17	User Define 15
18	User Define 16
19	User Define 17

10.7 HEAD TABLE VARIABLES

These variables allow to read and write parameters of the selected HEAD TABLE with **Hn**. The list below describes all parameters available for each HEAD TABLE and, as showed, they are up to maximum of 20 parameters per each head.

Tool table variable – ReadOnly

\$(Hn) Where **n** is a value from **0** to **19** indicating the parameter address

Ex:

\$VAR=\$(H0) **// READ OFFSET X**

Address	Descrizione
0	Head offset from home position of axis X
1	Head offset from home position of axis Y
2	Head offset from home position of axis Z
3	Head offset from home position of axis A
4	Head offset from home position of axis B
5	Head offset from home position of axis C
6	Head offset from home position of axis U
7	Head offset from home position of axis V
8	Sensor offset from 0 head of axis W
9	Sensor offset from 0 head of axis X
10	Sensor offset from 0 head of axis Y
11	Sensor offset from 0 head of axis Z
12	Sensor offset from 0 head of axis A
13	Sensor offset from 0 head of axis B
14	Sensor offset from 0 head of axis C
15	Sensor offset from 0 head of axis U
16	Sensor offset from 0 head of axis V
17	Sensor offset from 0 head of axis W
18	User Define 1
19	User Define 2

10.8 AXIS COUNTER VARIABLES

They are useful to read axis position from the interpolator. They don't contain the actual axis position but the position where it will be after position command. Generally these variable anticipate axis position before it has reached it. That is very useful for block restart facilities.

Axis counter variables – ReadOnly

\$(Cn) Where **n** is a value from **0** to **8** indicating the axis

11 LIMIT AXIS VARIABLES

They are useful to read the limit axis setted

Positive Limit – ReadOnly

\$[Nn] Where **n** is a value from **0** to **8** indicating the axis

Negative Limit – ReadOnly

\$[Sn] Where **n** is a value from **0** to **8** indicating the axis

11.1 GLOBAL VARIABLE

GLOBAL prefix can be applied only to GENERIC variables (\$name) and to use it in M or HM functions. GLOBAL prefix must be used when we want to share a variable between PartProgram and M/HM function, because by default IsoNs considers LABEL and VARIABLE private.

Ex:

```
//--- EXAMPLE FOR CODE OF PAUSE MACRO
//DEFINE $SAVEX AND $SAVEY AS GLOBAL
GLOBAL $SAVEX
GLOBAL $SAVEY
G62                // WAIT FOR AXIS IN POSITION
$SAVEX=$[Q0]      // SAVE AXIS X POSITION
$SAVEY=$[Q1]      // SAVE AXIS Y POSITION
G0X0Y0

//--- EXAMPLE FOR CODE OF RESTART FROM PAUSE MACRO
//DEFINE $SAVEX AND $SAVEY AS GLOBAL
GLOBAL $SAVEX
GLOBAL $SAVEY
G0X[$SAVEX] Y[$SAVEY] // MOVE AXIS TO POSITION SAVED IN PAUSE MACRO
G62
```

11.2 M FUNCTION VARIABLES (FOR CN)

These variable are useful to pass parameter to M functions defined in CN. That allows to condition M cycle by the value of them. The parameters must be write before to call M function. The number of passing parameters is defined in configuration up to maximum of 10 (default 5 parameters).

ATTENTION:

The CN get these parameters as integer value.

Ex:

```
// RUN M FUNCTION ON CN PASSING 2 PARAMETER
$_PARAM_1=100
$_PARAM_2=300
M15
```

11.3 SPECIAL PARAMETERS VARIABLES

They serve to read some parameters of IsoNs. The list below shows the readable parameters with these variables.

\$[X0]	Read the current value of SPEED selected by S function - <i>ReadOnly</i>
\$[X1]	Read the index of current WORK ORIGIN - <i>ReadOnly</i>
\$[X2]	Read the index of current WORK OFFSET - <i>ReadOnly</i>
\$[X3]	Read state of WORK ORIGIN – Value 1 if se enabled or 0 if disabled - <i>ReadOnly</i>
\$[X4]	Read state of WORK OFFSET – Value 1 if se enabled or 0 if disabled – <i>ReadOnly</i>
\$[X5]	Read the number of current head selected by H function – <i>ReadOnly</i>
\$[X6]	Read the number of current tool selected by T function – <i>ReadOnly</i>

\$[X7] Read The RUN Type – ReadOnly
 0 NORMAL RUN
 1 SIMULATION RUN
 2 GO BLOCK RUN
 3 RETRACE RUN

X8 Read Last FEED – ReadOnly

X9 Read coordinate mode – ReadOnly
 0 G90 Absolute
 1 G91 Relative

X10 Read work plane activated – ReadOnly
 0 XY G17
 1 XZ G18
 2 YZ G19
 -1 other

X11 Read tool length offset set – ReadOnly
 0 None
 1 Value (+/-) tool length offset set

X12 Read axis tool length offset set Legge – ReadOnly
 0 Axis X
 1 Axis Y
 2 Axis Z
 3 Axis A
 4 Axis B
 5 Axis C
 6 Axis U
 7 Axis V
 8 Axis W

11.4 WORK ORIGIN AND OFFSET VARIABLES

These variables contain the current position of WORK ORIGIN and WORK OFFSET. The maximum number of these variables is 9 (one for each axis). They can be used in PartProgram to read the current zero positions.

Work Origin Variable - ReadOnly

\$[Yn] Where **n** is a value from **0** to **8** indicating the axis. The variable will contain the current position of WORK ORIGIN of axis indicated by **n**

Work Offset Variable - ReadOnly

\$[Wn] Where **n** is a value from **0** to **8** indicating the axis. The variable will contain the current position of WORK OFFSET of axis indicated by **n**.

n=0 axis X, **n=1** axis Y, **n=2** axis Z, etc.

11.5 USER GENERIC VARIABLES

These variables are used to exchange data between CN and IsoNs. The maximum number available is 10. They can be read or written and are 32 bit INTEGER type.

User generic variables - Read/Write

\$[Kn] Where **n** is a value from **0** to **9** indicating the variable

11.6 ARRAY VARIABLES - DIM

These variables are used like to USER GENERIC VARIABLES, but these, are **get/set** by Index.

First to use, is necessary declare the ARRAY DIMENSION with Function DIM, and the VARIABLE ARRAY NAME:

Ex:

DIM \$ARR 10

In the above example, was declared an ARRAY with name ARR with 10 elements.

The ELEMENTS are indexed from 0 to 9

ARRAY USAGE

The ARRAY VARIABLES contain the same values of GENERIC VARIABLES (double).

For Get/Set the value from ARRAY, the INDEX value must be enclosed between "[..]".

Ex:

```
DIM $ARR 10
$INDEX=0
$VAR=2
LOOP 10
    $ARR[$INDEX]=$VAR*$INDEX
    $INDEX=$INDEX+1
END_LOOP
$INDEX=0
LOOP 10
    G1 X[$ARR[$INDEX]]
    $INDEX=$INDEX+1
END_LOOP
```

11.7 MARKER VARIABLES

Markers are normal variables written in PartProgram. In enhanced programming, using LOOP cycles and VARIABLES, restart from a LINE NUMBER is not enough, because axis position can be defined by variables written in a LOOP cycle. Using MARKERS it's possible restart PartProgram at value of one of these, not at a specific line number, allowing restart

also inside LOOP cycles. MARKERS are defined with instruction **MARKER \$NAMEVAR DESCRIPTION**.
Markers can be restored from PlugIn **"BLOCK RESTART"**

Ex:

In the next example it is defined a Marker variable named **\$INC**.

It specifies a counter of the number of piece in working

MARKER \$INC PIECE NUMBER

\$VAR=0

\$INC=0

F5

G1X0Y0

LOOP 10

\$INC=\$INC+1

G1X200

\$VAR=\$VAR+50

G0X0Y[\$VAR]

END_LOOP

It will be possible restart PartProgram when **\$INC** variable (set as MARKER) takes a specific value.

12 POSITIONER

IsoNs provides the complete management of one or more axes positioners. These can be of different type and in any case their management is entrusted to the application of CNC in use VTB

13 PA-PD(n,par)espr Set Absolute position

PA allows movement of the positioner at **ABSOLUTE** target

PD allows movement of the positioner at **RELATIVE** target

n Positioner NUMBER from 0 to MAX
par Controlo parameter OPTIONAL
null Start at TARGET **ESPR** with Pending end of movement
0 Start at TARGET **ESPR** without **Pending end of movement**
1 Start at TARGET **ESPR** with **Pending end of movement**
2 Start HOMING (Espr must be present but not used)
3 Espr >0 Enable positioner
 Espr <=0 Disable positioner

Ex:

```
PA(0)1000 // START AT TARGET 1000 POSITIONER 0
PA(0,0)1000 // START AT TARGET 1000 without pending end of movement
PA(0)[$VAR] // START AT TARGET$VAR i
PA(0,2)1 // START HOMING
PA(0,3)1 // ENABLE POSITIONER 0
```

14 PF(n)espr Set Positioner FEED

Set the FEED positioner. Espr FEED VALUE

n Positioner NUMBER from 0 to MAX

Ex:

```
PF(0)100 //FEED 100
PF(0)[$VAR] // FEED $VAR
```

15 PS(n) STOP Positioner

Stop POSITIONER

n Positioner NUMBER from 0 to MAX

Ex:

```
PS(0) // Stop positioner 0
```

16 _PM(n,par) Read status Positioner

Read the status positioner

n Positioner NUMBER from 0 to MAX

par Parameter

0 Return 0 Axis Stop fermo - 1 Axis movement
1 READ demand position
2 READ actual position
3 Return 1 axis enabled – 0 axis disabled
4 Return 1 HOMING make – 0 HOMING not make
5 Return 1 axis in ALLARM – 0 OK

Ex:

```
$VAR=_PM(0,par)
```

17 WORK SETTING

In this chapter are defined all axis motion functions.

17.1 DEFINITION OF AXIS POSITIONS

They define axis positions in the several interpolation functions (G0-G1-G2-G3). The axis code can be followed by a numeric value or by an expression identifying the position.

AXIS CODES

X	Axis X
Y	Axis Y
Z	Axis Z
A	Axis A
B	Axis B
C	Axis C
U	Axis U
V	Axis V
W	Axis W
I	Arc center for first axis of work plane
J	Arc center for first axis of work plane
R	Arc radius

Syntax

CODE value or expression

Ex:

```

X100Y20           // POSITION BY NUMERIC VALUE
DX100Y20          // RELATIVE AND ABSOLUTE
X[$VAR]Y[$VAR1]    // POSITION BY VARIABLES
X[cos($VAR)+$VAR1] // POSITION BY EXPRESSION
    
```

17.1.1 G90 – PROGRAMMING WITH ABSOLUTE POSITIONS

Syntax

G90

Function type

MODAL (default)

Cancel

G91

Description

All following positions refer to current ZERO (HOME POSITION, WORK ORIGIN, WORK OFFSET), If no work origin or offset are used positions refer directly to home position.

17.1.2 G91 - PROGRAMMING WITH INCREMENTAL POSITIONS

Syntax

G91

Function type

MODAL

Cancel

G90

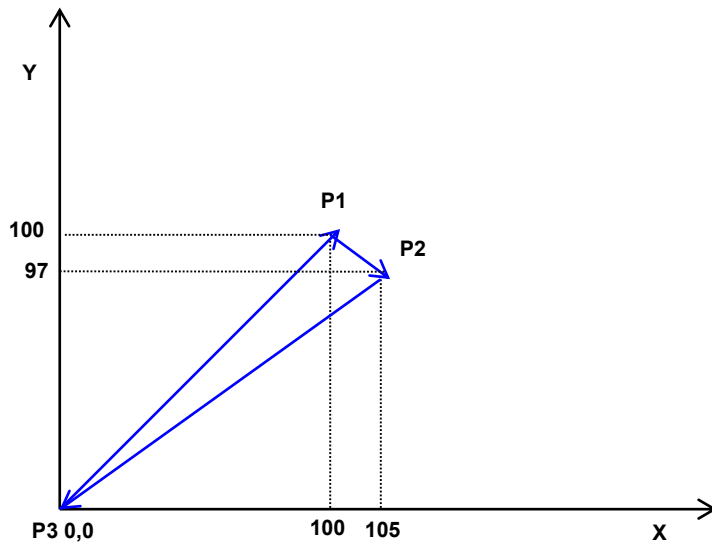
Description

All following positions refer to current axis position.

17.1.3 Definition of absolute and incremental positions

Ex:

```
G90          // SET ABSOLUTE POSITIONS
G1X100Y100   // AXIS MOVE TO P1=100,100
G91          // SET INCREMENTAL POSITIONS
X5Y-3        // AXIS MOVE TO P2=105,97
G90          // SET ABSOLUTE POSITIONS
X0Y0         // AXIS MOVE TO P3=0,0
```



17.2 WORK ORIGIN

ISONS can manage up to **256 different work origins** selectable by **G94**, **G92** and **USER_ZERO** instructions. A work origin defines a point of reference in relationship with home position.

17.2.1 Work origin By INDEX

These origins can be set on the fly by taking the value contained in 'file index' ZERI.VAL ". Unlike USER_ZERO, this can have different indexes for each axis.

Syntax

OXIndex **OY**Index

Function type

MODAL

Cancel

G92-G82

Parameters

Index –Index Value from file “ZERI.VAL”

Description

Defines the new origin of the axes based on the value contained in 'index set

Es:

```
OX0 OY1 OZ3 // SET origin X from INDEX 0, Y Index 1,Z Index 3
```

17.2.2 G94 – Work origin at position defined with parameter

Syntax

G94 Xvalue Yvalue (DX,DY ecc. Relative value)

Function type

MODAL

Cancel

G92-G82

Parameters

X,Y,Z,A,B,C,U,V,W followed by origin position

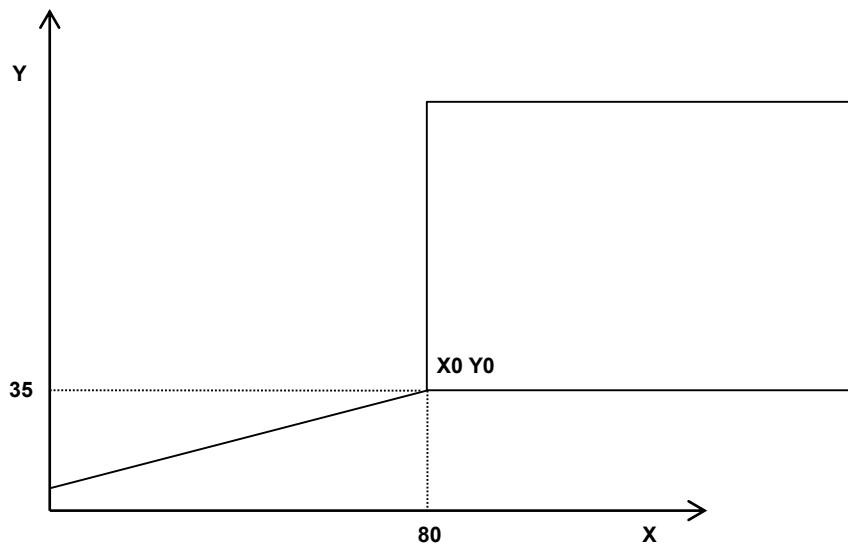
Description

It defines the new origin of one or more axis. Axis not included in parameters list keep the origin unchanged. The new position of zero axis will be defined in the parameter value.

The position value in the parameter refer to an ABSOLUTE position from HOME POSITION regardless of **G90 G91**. The new origin is automatically enabled with G94.

Ex:

G94 X80Y35 // SET NEW WORK ORIGIN AT 80,35



17.2.3 G54, G55, G56, G57, G58, G59 - Work origin from memory file

Syntax

G54 (axis code)

G54 XYZ or G54.n XYZ

Function type

MODAL

Cancel

G92-G82

Parameters

X,Y,Z,A,B,C,U,V,W

.n Number of INDEX ZERO (from 0 to 255). If omitted, uses the parameter **USER_ZERO** to set index

Description

It defines the new origin of one or more axis. Axis not included in parameters list keep the origin unchanged. The new position of zero axis will be defined in the parameter value.

The position value will be get from the ORIGIN FILE made by **PLUGIN WORK ORIGIN**.

The new origin is automatically enabled with these instructions.

If **G54** is invoked without parameters **AXIS X** will be set.

The **G54** function, can use the INDEX ZERO indicate in the **.n**

G54.n where **n** is a value from 0 to 255 refered to INDEX ORIGIN

If the parameter **.n** is omitted, is used **USER_ZERO** to set index.

The **.n** parameter not changes the value of **USER_ZERO** parameter.

Similarly: (these can not use the **.n** parameter)

G55 for **AXIS Y**

G56 for **AXIS Z**

G57 for **AXES XY**

G58 for **AXES YZ**

G59 for **AXES XZ**

Unlike G94, these functions set WORK ORIGIN from a file saved on pc allowing reload of ORIGINS after switch-off.

Ex:

USER_ZERO 0 // uses INDEX 0 (Option default)

G54 XYZ // Set ORIGIN to X Y Z

USER_ZERO 1 // uses INDEX 1

G55 // Set origin Y

G54.2 XYZ // SET Origin on XYZ by INDEX 2

17.2.4 G92 – Work origin in current axis position

Syntax

G92 XYZ etc.

Function type

MODAL

Cancel

G94-G82

Parameters

X,Y,Z,A,B,C,U,V,W axis to be zero

Description

It defines the new origin of one or more axis in the current position. Axis indicated are zeroed in current position.

17.2.5 G82 – Work origin in current axis position with sensor offset

Syntax

G82 XYZ etc.

Function type

MODAL

Cancel

G94-G92

Parameters

X,Y,Z,A,B,C,U,V,W axis to be zero

Description

It defines the new origin of one or more axis in the current position taking into account acquisition sensor (G102). Axis indicated are zeroed in current position adding sensor offset.

This function must be used with origin acquisition cycles by G102.

17.2.6 G98 – G53 - Suspend work origin

Syntax

G98

G53

Function type

MODAL

Cancel

G82 G92 G94 -G54-G55-G56-G57-G58-G59

Description

It suspends all work origins. All following positions will be refer to HOMING POSITION (if WORK OFFSETS G93 or G95 are disabled)

17.2.7 G99 – Restore work origin

Syntax

G99

Function type

MODAL

Cancel

G98

Description

Restore work origin set with G92 or G94, disabling G98.

17.2.8 G940 MOVE axes excluding WORK ORIGIN only in the actual block

Syntax

G940 **G1**Xval Yval

Function type

deleting

Description

All axis movements follow G940 are performed with reference to home axes (WORK ORIGIN EXCLUDES)

17.2.9 USER_ZERO – Index of WORK ORIGIN LIST

Syntax

USER_ZERO expression

Function type

MODAL

Parameters

Expression or value indicating the index of ORIGIN LIST to be used

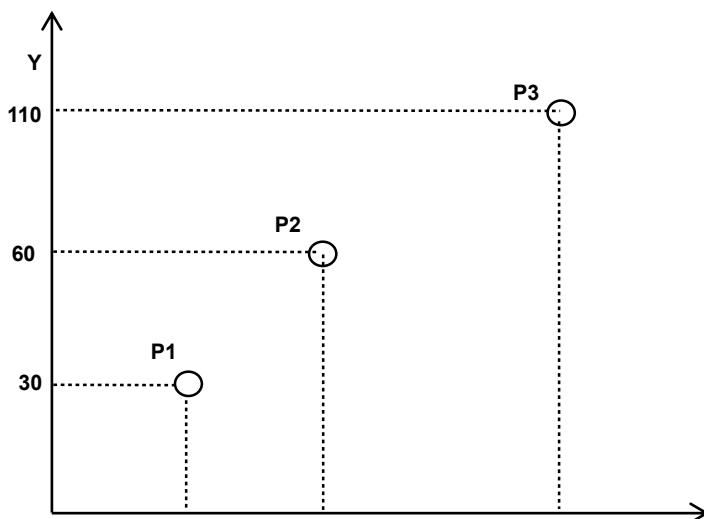
Description

Work origin is enabled with the position indexed in the ORIGIN LIST by USER_ZERO.

Ex:

```

USER_ZERO 0
G94 X20Y20           // STORE ORIGIN 0
USER_ZERO 1
G94 X50Y50           // STORE ORIGIN 1
USER_ZERO 2
G94 X150Y100        // STORE ORIGIN 2
USER_ZERO 0             // ENABLE ORIGIN 0
G1X10Y10              // MOVE TO P1
USER_ZERO 1             // ENABLE ORIGIN 1
G1X10Y10              // MOVE TO P2
USER_ZERO 2             // ENABLE ORIGIN 2
G1X10Y10              // MOVE TO P3
    
```



17.3 WORK OFFSET

IsoNs can manage up to **256 different work offsets** selectable by **G93**, **G95** and **USER_OFFSET** instructions. A work offset is an additional work origin respecting G92 and G94 functions. As the last one these define a point of reference in relationship with home position. For example see WORK ORIGIN.

17.3.1 G93 - Work offset at position defined with parameter

Syntax

G93 Xvalue Yvalue (DX,DY ecc. Relative value)

G93 P(n)value DP(n)value for positioner

Or **DOX**value **DOY**value

Function type

MODAL

Cancel

G85 G95

Parameters

X,Y,Z,A,B,C,U,V,W followed by offset position

Description

It defines the new offset of one or more axis. Axis not included in parameters list keep the offset unchanged. The new position of zero axis will be defined in the parameter value.

The position value in the parameter refer to an ABSOLUTE position from HOME POSITION regardless of **G90 G91**. The new offset is automatically enabled with G93.

17.3.2 G95 - Work offset in current axis position

Syntax

G92 XYZ etc.

Function type

MODAL

Cancel

G85 G93

Parameters

X,Y,Z,A,B,C,U,V,W axis to be zero

Description

It defines the new offset of one or more axis in the current position. Axis indicated are zeroed in current position.

17.3.3 G85 - Work offset in current axis position with sensor offset

Syntax

G85 XYZ etc.

Function type

MODAL

Cancel

G93-G95

Parameters

X,Y,Z,A,B,C,U,V,W axis to be zero

Description

It defines the new offset of one or more axis in the current position taking into account acquisition sensor (G102). Axis indicated are zeroed in current position adding sensor offset.

This function must be used with origin acquisition cycles by G102.

17.3.4 G86 - Hardware preset axis on module 360 degrees

Syntax

G85 XYZ etc.

Function type

IMMEDIATE

Parameters

X,Y,Z,A,B,C,U,V,W axis to be preset

Description

The G86 function, is used for rotative axes.

It preset in hardware mode, the absolute axis position to the relative angle.

Ex:

I position axis is 1.450, the relative degrees is 10

17.3.5 G96 - Suspend work offset

Syntax

G96

Function type

MODAL

Cancel

G85 G93 G95

Description

It suspends all work offset. All following positions will be refer to HOMING POSITION (if WORK ORIGINS G92 or G94 are disabled).

17.3.6 G97 - Restore work origin

Syntax

G97

Function type

MODAL

Cancel

G96

Description

Restore work offset set with G93 or G95, disabling G96.

17.3.7 USER_OFFSET - Index of WORK OFFSET LIST

Syntax

USER_OFFSET expression

Function type

MODAL

Parameters

Expression or value indicating the index of OFFSET LIST to be used

Description

Work offset is enabled with the position indexed in the OFFSET LIST by USER_OFFSET.

Ex:

```

USER_OFFSET 0
G94 X20Y20           // STORE OFFSET 0
USER_OFFSET 1
G94 X50Y50           // STORE OFFSET 1
USER_OFFSET 2
G94 X150Y100        // STORE OFFSET 2
USER_OFFSET 0 // ENABLE OFFSET 0
G1X10Y10            // MOVE TO P1
USER_OFFSET 1 // ENABLE OFFSET 1
G1X10Y10            // MOVE TO P2
USER_OFFSET 2 // ENABLE OFFSET 2
G1X10Y10            // MOVE TO P3

```

Refer to ORIGIN example for drawing.

17.4 Work head selection – H function

ISOs can manage up to **256 different tools heads**.

Each head refer to home position by the value in the HEADS TABLE.

By default no head is enabled, with Hn function it's possible to select one of 256 available heads (first head =0). All offset (one for each axis) of the selected head will be automatically enabled.

17.4.1 Hn – Select head

Syntax

H expression

Function type

MODAL

Parameters

Expression or value identifying selected head (first head 0)

Description

Offset of the selected head are enabled.

Ex:

H0 // SELECT FIRST HEAD

17.4.2 G87 – Suspend head offset

Syntax

G87

Function type

MODAL

Cancel

G88

Description

Offset of selected head are disabled.

Ex:

G87 // SUSPEND HEAD OFFSET

17.4.3 G88 – Restore head offset

Syntax

G88

Function type

MODAL

Cancel

G87

Description

Last enabled head offset are restored.

Ex:

G88 // RESTORE HEAD OFFSET

17.5 ROTATIVE AXIS

IsoNs can manage rotative axis in several modes. G36 function selects mode.

17.5.1 G36 – Rotative axis definition

Syntax

G36 Xmode AXISNAME

Function type

MODAL

Parameters

Mode: 0 → Disable (set linear axis)

1 → Shorter path.

2 → Always positive

3 → Always negative

4 → Shorter path when G1, disabled when G2/G3

AXISNAME name of rative axis (ex A,B etc.)

Description

Mode 1,4 - Shorter path

CN chooses the shorter path to reach the position selected.

Mode 4 is similar to mode 1 but in G2/G3 interpolation this facilities is disabled.

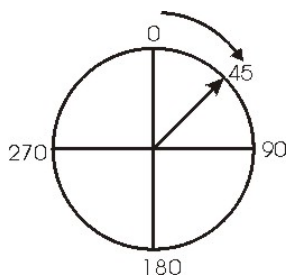
Ex:

G36 X1 A

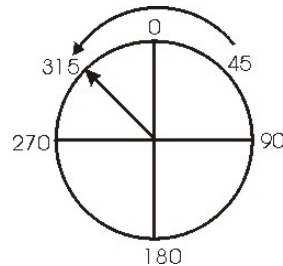
G1 A45

G1 A315

first movement **G1 A45**



second movement **G1 A315**



Mode 2,3 – Always positive/negative

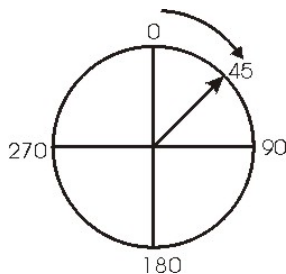
The selected axis rotates always in the same sense.

G36 X2 A

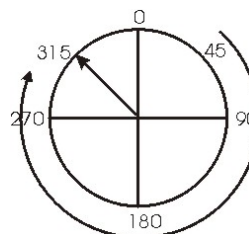
G1 A45

G1 A315

first movement **G1 A45**



second movement **G1 A315**



17.6 HARDWARE PRESET OF AXIS

In particular cases can be necessary preset axis positions in Hardware mode allowing to avoid Overflow in single direction axis.

Hardware preset must be used with particular attention, because if axis works with a following error, it doesn't take into consider. Consequently it's possible to lost encoder pulses and reference to home position.

17.6.1 G89 – Hardware preset of axis position

Syntax

G89 Xvalue Yvalue (DX,DY ecc. Relative value)

Function type

DIRECT

Cancel

All origin and offset functions

Parameters

X,Y,Z,A,B,C,U,V,W followed by preset value

Description

It defines a new value for current axis positions.

17.7 HARDWARE PRESET OF AXIS

In some cases it is necessary to preset the counters of the axes after recovering from a PAUSE, or to RETRACE. This is because during the PAUSE or first RETRACE, the axes are moved using special functions (shift-type etc..).

Not properly updating the counters, abnormal movements may occur during the step of interpolation axes. Generally counters axes are always updated and therefore there is no need an update, but there are cases such as those listed above that require an update. Always use an upgrade at the end of the axis M of recovery from the PAUSE by RETRACE is still a good thing .

17.7.1 G84 – Preset Axes counters

Syntax

G84 X Y Z....

Function type

DIRECT

Parameters

X,Y,Z,A,B,C,U,V,W axes to update

Unless indicated no axis are updated all axes

Description

The G84 function copies the values of the coordinates axes by process (typically taken from PAUSE or RETRACE) the process of WORK.

17.7.2 G83 – Preset CPU 1 counters

Syntax

G83

Function type

DIRECT

Description

The G83 function is only used to force the update of the counters of the CPU 1, the one that works on MACRO PAUSE, STOP, ERROR.

it is advisable to start with the MACRO code G83

17.8 SELECT WORK PLANE

Work plane defines:

- The plane for circular interpolation
- The plane where to calculate the axis stop (fast interpolation G60)
- The plane where to calculate the radius tool compensation

17.8.1 G17 – Select work plane on X-Y

Syntax

G17

Function type

MODAL

Cancel

G18 G19 G70

17.8.2 G18 - Select work plane on X-Z

Syntax

G18

Function type

MODAL

Cancel

G17 G19 G70

17.8.3 G19 - Select work plane on Y-Z

Syntax

G19

Function type

MODAL

Cancel

G17 G18 G70

17.8.4 G70 - Select work plane on axis by parameters

Syntax

G70 AXIS1 AXIS2

Function type

MODAL

Cancel

G17 G18 G19

Parameters

Name of the couple of axis formed the new work plane

Ex:

G70XA *// select work plane on axis X and A*

17.9 START SEARCH OF HOME POSITION

With G71 function IsoNs allows to execute searching of home position directly in PartProgram. It can be used, for example, when there are spindle axis with double function:

SPEED AXIS

CONTROLLED AXIS

In these cases when we switch function mode to a controlled axis it will be necessary to make searching of home position.

17.9.1 G71 – Start Homing cycle

Syntax

G71 ASSE

Function type

DIRECT

Parameter

Name of axis to be homed

Description

It starts searching of home position for selected axis. Searching mode is defined by machine parameter RZERO_MODE. PartProgram waits for ending of homing procedure.

However STOP PROGRAM is always active. By the parameter of configuration TIME_OUT_HOME can be generated an error (1005 Time Out axis Home) if searching isn't terminated in the set time.

Ex:

G71 X

17.10 AXIS MOVEMENT FUNCTIONS

17.10.1 G0 – Rapid positioning

Syntax

G0

Function type

MODAL

Cancel

G1 G2 G3

Description

G0 defines linear movement type with maximum axis speed.

Maximum speed of each axis is defined by machine parameters.

IsoNs use a specific calculation of axis speed reducing to minimum the axis STRESS during rapid positioning. Unlike other CN where G0 defines **NOT-INTERPOLATED** movement, IsoNs move axis always in **INTERPOLATION** mode, but calculating the best speed. In this mode we can obtain always a **FLUID MOTION** where all axis start and arrive simultaneously.

Furthermore it is used a specific **ACCELERATION** parameter different from the working one. It is recommended to use G0 only for movement with tool not in working.

Ex:

G0X0Y0Z0 // MOVE TO 0,0,0 POSITION

17.10.2 G1 – Linear interpolation a programmed F speed

Syntax

G1

Function type

MODAL

Cancel

G0 G2 G3

Description

G1 defines linear interpolation mode at programmed speed. If movement are on selected work plane and it's enabled **FAST interpolation (without stop on edge) G60**, IsoNs calculates automatically stop on edge. G1 enables movement at programmed F speed and can be modified by override potentiometer (if not disabled with G11). According to CN we can obtain different performance about number of segment processed in a time unit. Cn of NG35 series process about **350 segments per second**, instead **120** for CN of NGM13 series. That defines the maximum working velocity when the path is formed by **micro-segments**. All CN have a movement buffer, IsoNs tries to maintain it full (in G60 mode). If buffer empties CN stops axis obtaining a not fluid movement.

Ex:

G1X300Y200Z20

17.10.3 G2/G3 - Circular interpolation a programmed F speed

Syntax

G2 (cw)

G3 (ccw)

Function type

MODAL

Cancel

G0 G1

Description

G2 and G3 define circular interpolation at programmed speed. If movement are on selected work plane and it's enabled **FAST interpolation (without stop on edge) G60**, IsoNs calculates automatically stop on edge. G2 and G3 enable movement at programmed F speed and can be modified by override potentiometer (if not disabled with G11).

Circular interpolation can be executed by center parameters I and J, or by radius parameter R. With the second one it's not possible to generate a complete circle, for that it must be used I and J.

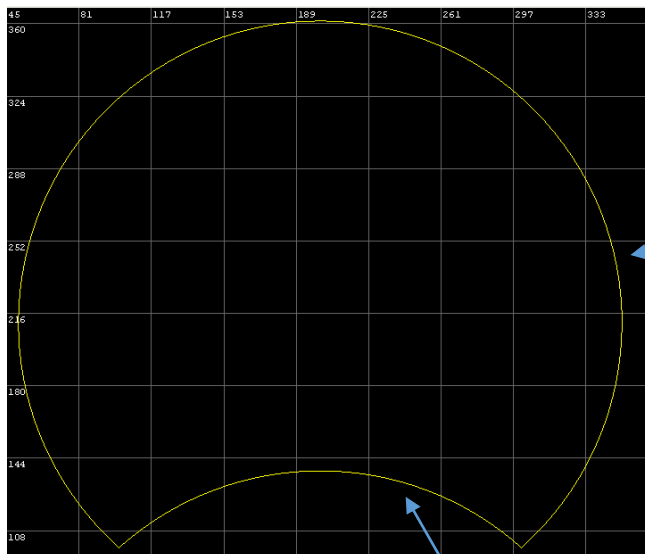
SHORT WAY and LONG WAY arc

Using radius parameter there are two arcs defined by the same parameters. The selection is via radius sign.

If RADIUS is positive a SHORT WAY ARC is generated.

If RADIUS is negative a LONG WAY ARC is generated.

See the example for details.



LONG WAY ARC
R-150

SHORT WAY ARC
R150

Ex:

G0X100Y100

G2X300Y100R-150 // EXECUTE LONG WAY ARC

G0X100Y100

G2X300Y100R150 // EXECUTE SHORT WAY ARC

17.10.4 G30 - Enable automatic insert of fillet on edges

Syntax

G30 **R**radius **A**angle

Function type

MODAL

Cancel

G33

Parameters

R value of fillet radius (radius=0 disables the function)

A Angle degree of insert threshold

Description

G30 inserts automatically fillets on edges. The fillet radius is defined with parameter R, parameter A defines the angle threshold under which G30 function doesn't intervenes leaving path unchanged. This function is useful to soften movement in angular path.

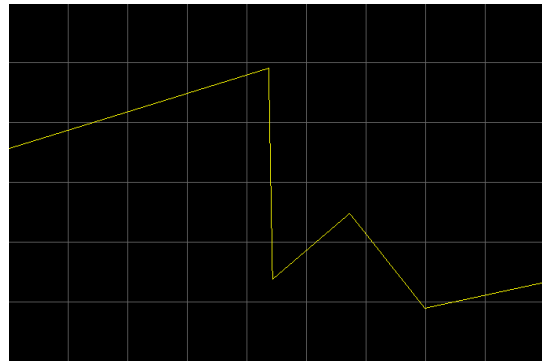
Fillet is inserted only if it can be container inside two segment.

Setting a RADIUS=0 the function is disabled.

Ex:

// NORMAL PATH WITHOUT G30

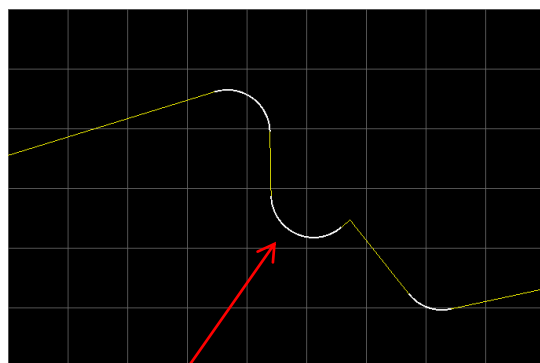
```
G0 X75.45 Y157.46
G1 X107.84 Y167.43
G1 X108.3 Y142.73
G1 X117.36 Y150.44
G1 X126.2 Y139.34
G1 X140.92 Y142.51
```



Ex:

// SAME PATH USING G30

```
G0 X75.45 Y157.46
G30 R5 A20 // RADIUS=5 ANGLE=20 deg.
G1 X107.84 Y167.43
G1 X108.3 Y142.73
G1 X117.36 Y150.44
G1 X126.2 Y139.34
G1 X140.92 Y142.51
```



R=5 mm

G31 – Suspend of automatic insert of fillet

Syntax

G31

Function type

MODAL

Cancel

G30

Description

Suspend G30. It can be restore with G32.

17.10.5 G32 - Restore of automatic insert of fillet

Syntax

G32

Function type

MODAL

Cancel

G31

Description

Restore G30 suspended by G31.

17.10.6 G33 - Enable automatic insert of bevel on edges

Syntax

G33 Rlen Aangle

Function type

MODAL

Cancel

G30

Parameter

R *length of bevel (len=0 disables the function)*
A *Angle degree of insert threshold*

Description

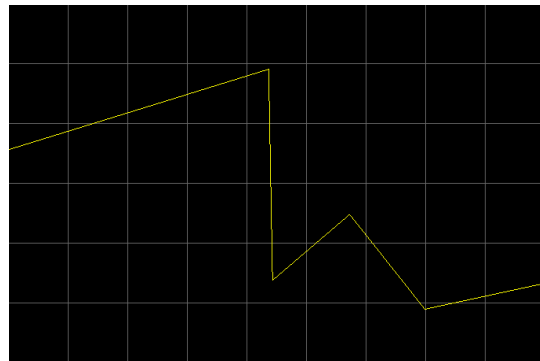
G33 inserts automatically bevels on edges. The bevel length is defined with parameter R, parameter A defines the angle threshold under which G33 function doesn't intervenes leaving path unchanged. This function is useful to soften movement in angular path.

Bevel is inserted only if it can be container inside two segment.

Setting a **RADIUS=0** the function is disabled.

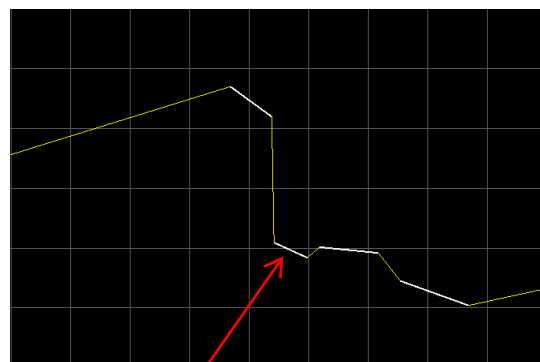
Ex:

```
// NORMAL PATH WITHOUT G33
G0 X75.45 Y157.46
G1 X107.84 Y167.43
G1 X108.3 Y142.73
G1 X117.36 Y150.44
G1 X126.2 Y139.34
G1 X140.92 Y142.51
```



Ex:

```
// SAME PATH USING G33
G0 X75.45 Y157.46
G33 R5 A20 // LEN=5MM ANGLE=20 deg.
G1 X107.84 Y167.43
G1 X108.3 Y142.73
G1 X117.36 Y150.44
G1 X126.2 Y139.34
G1 X140.92 Y142.51
```



Len=5 mm

17.10.7 G34 - Suspend of automatic insert of bevel

Syntax

G34

Function type

MODAL

Cancel

G33

Description

Suspend G33. It can be restore with G35.

17.10.8 G35- Restore of automatic insert of bevel

Syntax

G35

Function type

MODAL

Cancel

G34

Description

Restore G33 suspended by G34.

17.10.9 G102 – Start searching sensor position

Syntax

G102

Function type

DIRECT

Description

It start acquisition from sensor.

Axis move to positions selected by X,Y,Z etc. The values refer to an absolute or relative position according with G90 G91. Selected **WORK ORIGIN** and **WORK OFFSET** are considered too. More axis can be move simultaneously (interpolated) and the speed is set with the machine parameter **ACQ_VEL**.

Also a acquisition mode can be selected by machine parameter **ACQ_MODE**.

Executing can be stopped by activation of **SENSOR** or by reaching **TARGET** position. In the first case PartProgram continues to execute from next block to G102.

In case of reaching of **TARGET** position PartProgram ends with **ACQ. SENSOR ERROR**.

G102 waits for axis stop before to proceed with acquisition cycle.

Ex:

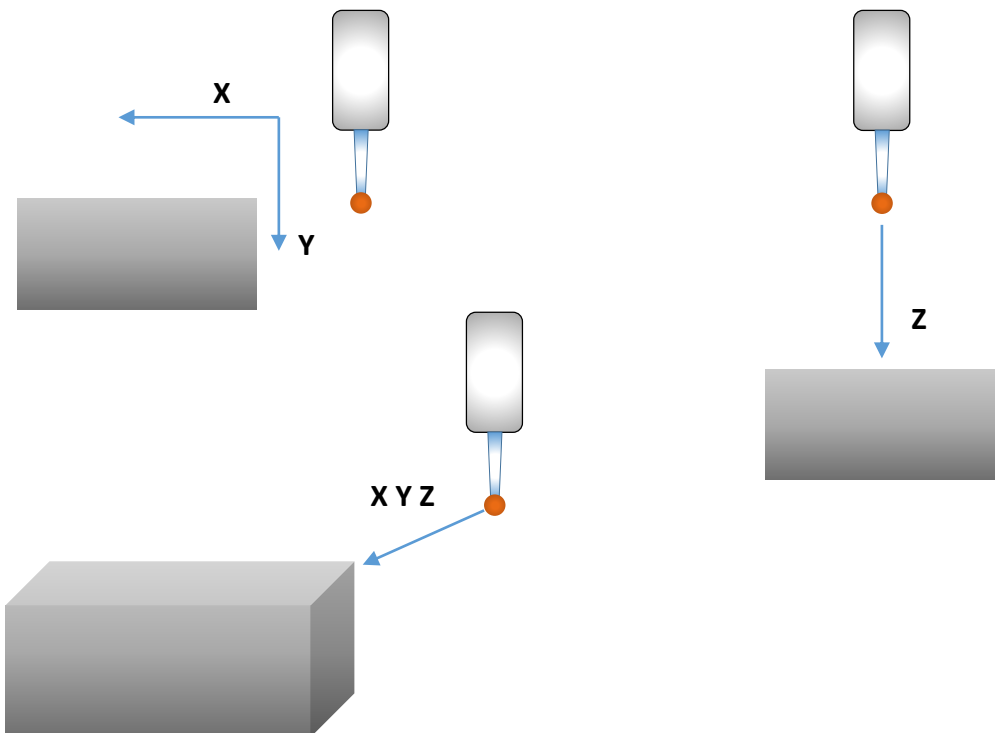
G90

G102 X100

G92 X

// ACQUISITION MOVING ONLY AXIS X UP TO POSITION 100

// SET WORK ORIGIN IN ACQUIRED POINT



17.11 PROGRAMMING AXIS INTERPOLATION SPEED

Interpolation speed is a very important parameter to obtain a fine working. Speed value is expressed in Mt/min with a resolution dependent by IsoNs configuration. Selected interpolation speed can be changed in automatic way (arc auto-reduction), or manually with parameter **F** or override potentiometer. F parameter acts only when invoked in PartProgram and however at begin of next segment, OVERRIDE potentiometer instead acts in any time allowing immediately change of speed.

17.11.1 F – Speed of axis interpolation

Syntax

Fval

Function type

MODAL

Parameter

Val *Value in Mt/min of speed*

Description

F function set vectorial speed of the axis. Such speed will be separated by interpolator according with the programmed trajectory.

G0 movement are not affected by parameter F.

Ex:

```
G0 X75.45 Y157.46      // RAPID MOVEMENT AT F MAX
F10.3                  // SET AXIS SPEED AT 10.3 MT/MIN
G1 X107.84 Y167.43
```

17.11.2 ARC speed auto-reduction

Function type

Configuration programming

Parameter

See IsoNs configuration

Description

IsoNs uses a special algorithm which allows to reduce automatically axis speed during circular interpolation. It simplifies F programming in path formed by arcs without to insert an appropriate speed for each arc (lower speed for lower radius – bigger speed for bigger radius), IsoNs does it for you.

The calculation is base on the centrifugal acceleration and it must be manually searched experimentally .

17.12 Work Plane Transformation

Here they are explained all function relative to manage with work plane such as: rotation and mirror.

17.12.1 G120 – Vertical mirror

Syntax

G120

Function type

MODAL

Cancel

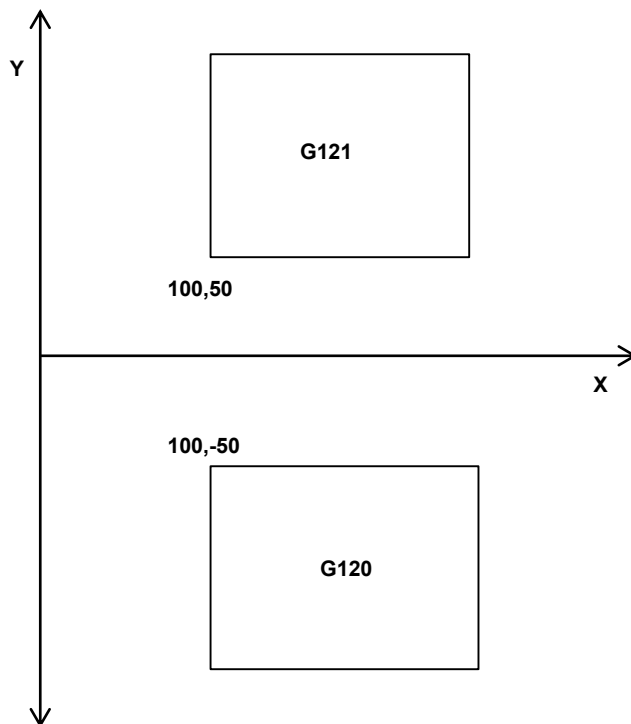
G121

Description

It enables vertical mirror on selected work plane. It acts inverting automatically the sign of axis Y.

Ex:

```
G121 // DISABLE G120
// ***** EXECUTE FIRST SQUARE
G0 X100 Y100
G1 X150
Y50
X100
Y100
// ***** END FIRST SQUARE
G120 // ENABLE VERTICAL MIRROR
// ***** EXECUTE SECOND SQUARE
G0 X100 Y100
G1 X150
Y50
X100
Y100
// ***** END SECOND SQUARE
```



17.12.2 G121 – Disable vertical mirror

Syntax

G121

Function type

MODAL

Cancel

G120

Description

Disable vertical mirror on work plane.

17.12.3 G24 – Horizontal mirror

Syntax

G24

Function type

MODAL

Cancel

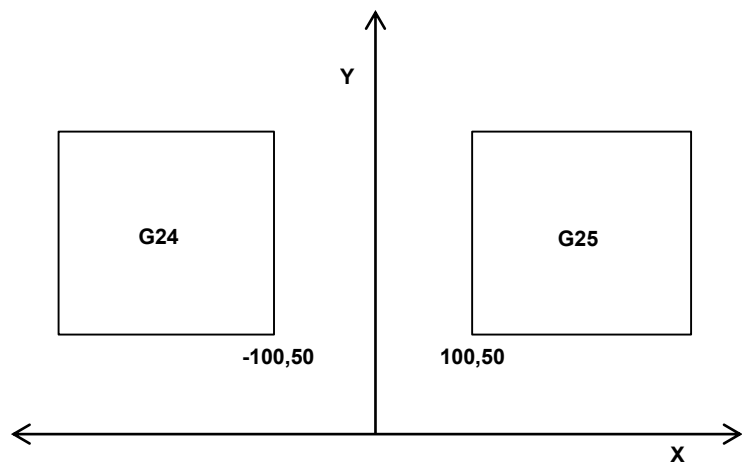
G25

Description

It enables horizontal mirror on selected work plane. It acts inverting automatically the sign of axis X.

Es:

```
G25 // DISABLE G24
// ***** EXECUTE FIRST SQUARE
G0 X100 Y100
G1 X150
Y50
X100
Y100
// ***** END FIRST SQUARE
G24// ENABLE VERTICAL MIRROR
// ***** EXECUTE SECOND SQUARE
G0 X100 Y100
G1 X150
Y50
X100
Y100
// ***** END SECOND SQUARE
```



17.12.4 G25 – Disable horizontal mirror

Syntax

G25

Function type

MODAL

Cancel

G24

Description

Disable horizontal mirror on work plane.

17.12.5 G22 – Exchange axis of work plane

Syntax

G22

Function type

MODAL

Cancel

G23

Description

Axis of work plane are exchanged between them (ex. X with Y).

17.12.6 G23 – Restore axis of work plane

Syntax

G23

Function type

MODAL

Cancel

G22

Description

Restore the original axis of work plane.

17.12.7 G26 – Exchange axis by parameter

Syntax

G26 Axis1 Axis2

Function type

MODAL

Cancel

G27

Description

A couple of selected axis by parameters are exchanged between them.

Ex:

G26 YZ // EXCHANGE Y WITH Z

17.12.8 G27 – Suspend G26

Syntax

G27

Function type

MODAL

Cancel

G26

Description

It suspends the axis exchanging set with G26

17.12.9 G28 – Restore G26

Syntax

G28

Function type

MODAL

Cancel

G27

Description

Restore axis exchanging set with G26

17.12.10 ***G51 – Suspend work plane rotation***

Syntax

G51

Function type

MODAL

Cancel

G50 G52

Description

Suspend the rotation of work plane.

17.12.11 ***G52 – Restore work plane rotation***

Syntax

G52

Function type

MODAL

Cancel

G51

Description

Restore last work plane rotation set with G50.

17.12.12 G50 - Work plane rotation

Syntax

G50 **X**centerx **Y**centery **Z**angle (DX,DY Relative value)

Function type

MODAL

Cancel

G51

Parameters

X *Rotation center axis X*
Y *Rotation center axis Y*
Z *Rotation angle in degree*

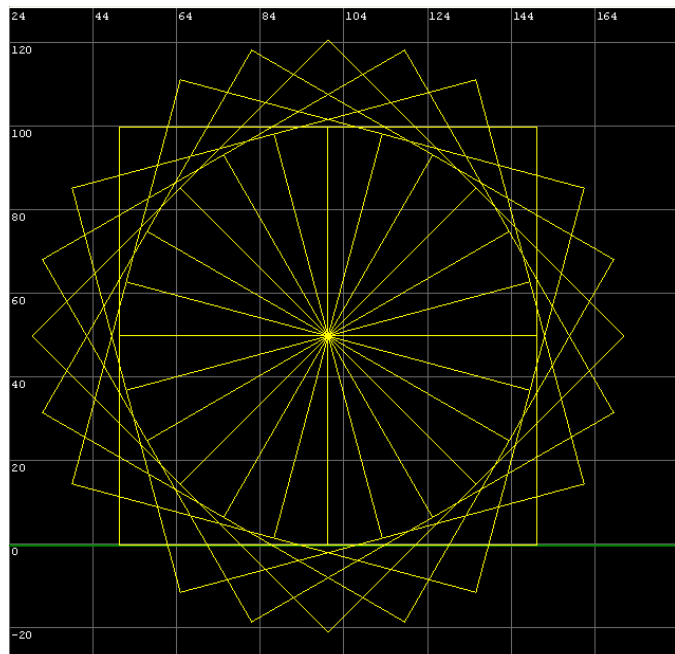
Description

It execute a rotation of work plane. All following position will be modified with this transformation.

At each start of PartProgram rotation center is disabled.

Ex:

```
// ROTATION OF A SQUARE
$ROT=0
LOOP 24
G0 X100 Y100
G1 X150
Y50
X100
Y100
$ROT=$ROT+15// INCR. ANGLE
G50 X100 Y50 Z[$ROT]// ROTATION
END_LOOP
```



17.12.13 G1050 – Disable Scaling

Syntax

G1050

Function type

MODAL

Cancel

G1051

Description

G1050 function, disable SCALING G1051

17.12.14 G1051 – Enable Scaling

Syntax

G1051 Xcentrox **Y**centroy **Z**centroz **I**ratioX **J**ratioy **K**ratioz **P**ratioxyz

Function type

MODAL

Cancel

G1050

Parameters

X	<i>X Scaling Center – se non inserito viene presa la posizione attuale dell' asse X</i>
Y	<i>Y Scaling Center – se non inserito viene presa la posizione attuale dell' asse Y</i>
Z	<i>Z Scaling Center – se non inserito viene presa la posizione attuale dell' asse Z</i>
I	<i>X Ratio – X<1 reduces X>1 increases - if it is not inserted, uses the previous value</i>
J	<i>Y Ratio – Y<1 reduces Y>1 increases - if it is not inserted, uses the previous value</i>
K	<i>Z Ratio – Z<1 reduces Z>1 increases - if it is not inserted, uses the previous value</i>
P	<i>XYZ Ratio – P<1 reduces P>1 increases – Uses the same Ratio for X,Y,Z</i>

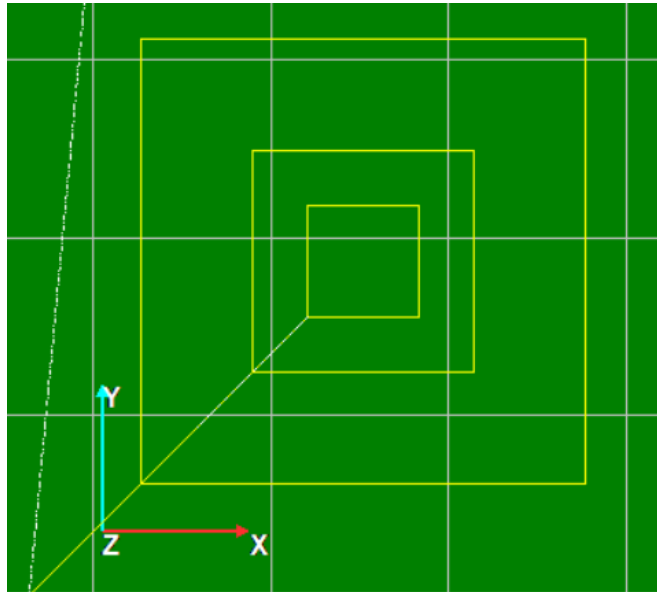
IF NO PARAMETERS ARE INSERTED (Only G1051) the previous values are used

Description

G1051 allows to scaling the Gcode with X,Y,Z Coordinate center.

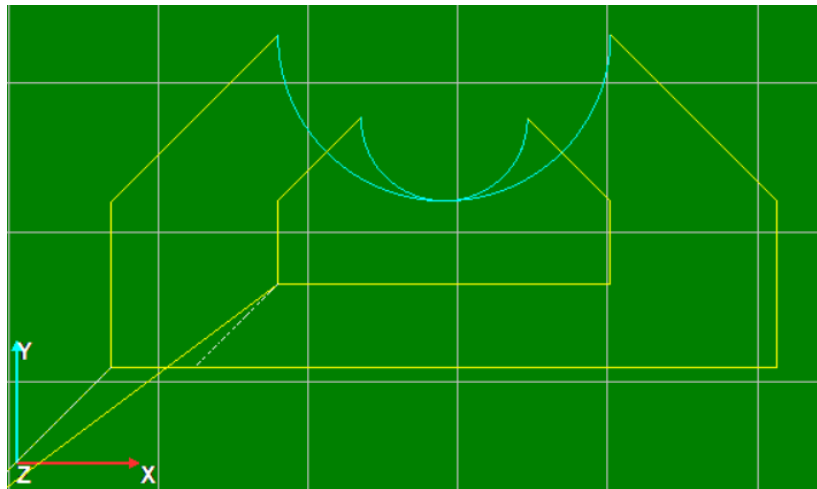
Ex:

```
//ORIGINAL
G0X0Y0
G1X10Y10
Y20
X20
Y10
X10
G0X0Y0
//SCALING
G1051 X15Y15I0.5J0.5
G1X10Y10
Y20
X20
Y10
X10
G0X0Y0
//SCALING
G1051 X15Y15I2J2
G1X10Y10
Y20
X20
Y10
X10
```



Ex:

```
//ORIGINAL
G0X0Y0
G1X10Y10
Y20
X20Y30
G3X40Y30R10
G1X50Y20
X50Y10
X10Y10
G0X0Y0
//SCALING
G1051 X30Y20I0.5J0.5
G1X10Y10
Y20
X20Y30
G3X40Y30R10
G1X50Y20
X50Y10
X10Y10
G0X0Y0
```



17.12.15 *G103 – Set RTCP parameters*

Syntax

G103 Xval Yval Zval

Function type

MODAL

Parameters

X *Value of parameter X*

Y *Value of parameter Y*

Z *Value of parameter Z*

.....

Description

ISONs uses a **RTCP** (rotate tool center point) adaptable to any geometry of the machine. Therefore the value passed to function by G103 (max 9 value X,Y,Z,A,B,C,U,V,W) will determine the right parameters for a specific geometry. The calculation of RTPC must be set in installation phase and it will be necessary the intervention of a specialized technician. **G103** passes the parameters to CN without enabling RTCP function. Enable/disable is demanded to **G104-G105** function.

17.12.16 *G104 – Enable/restore RTCP*

Syntax

G104

Function type

MODAL

Cancel

G105

Description

G104 enable or restore **RTCP** function on CN.

17.12.17 *G105 – Disable RTCP*

Syntax

G105

Function type

MODAL

Cancel

G104

Description

G105 disable **RTCP** function on CN restoring normal interpolation. RTCP function can be restored with **G104**.

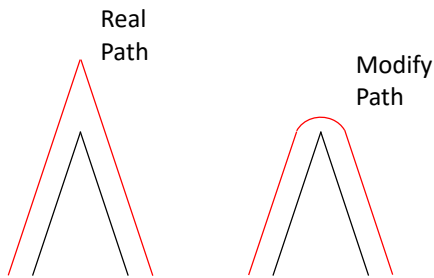
18 TOOL RADIUS COMPENSATION

ISONS manages the tool offset compensation, that can be activated with G41 and G42 ISO function. This functions allow to adjust the tool track, according the tool diameter. Using this mode, it's possible to have some interference on tool track, if we have high diameter tools.

ISONS can display on its interface, a graphic preview, where will be showed interfering points, due to make possible to adjust the track, previous the work start.

However the tool compensation will take high attention on the track elaboration.

Furthermore it will insert automatically connection arcs (with tool radius as arc radius) , if the intersection point of the compensation tracks, is very far away from real intersection point.



18.1.1 G41/G42 – Enable left/right radius tool compensation

Syntax

G41

Function type

MODAL

Cancel

G40

Description

G41 Enable left compensation.

G42 Enable right compensation.

Tool diameter is set by **D** function or by tool table **Tn**.

18.1.2 G40 – Disable radius tool compensation

Syntax

G40

Function type

MODAL

Cancel

G41 - G42

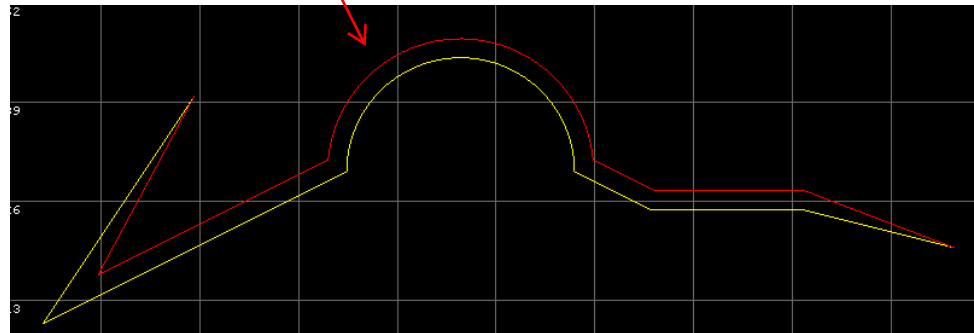
Description

G40 disables the radius tool compensation (both left and right).

PATH WITH G41

Ex:

```
D5// DIAMETER=5mm
G41// LEFT COMP.
G0X30Y40
G1X10Y10
G1X50Y30
G2X80Y30R15
G1X90Y25
G1X110
G1X130Y20
G40
```

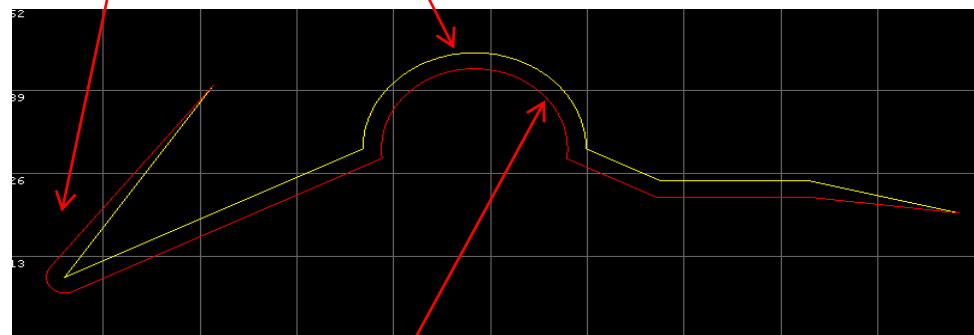


PATH WITH G40

ADDING ARC

Ex:

```
D5// DIAMETER=5mm
G42// RIGTH COMP.
G0X30Y40
G1X10Y10
G1X50Y30
G2X80Y30R15
G1X90Y25
G1X110
G1X130Y20
G40
```



PATH WITH G42

18.1.3 D – Tool diameter

Syntax

Dvalue

Function type

MODAL

Cancel

Tn (optional correction set by TOOL TABLE)

Parameter

Value Tool diameter in mm

Description

D allows to set the value in mm of tool diameter to be used with G41 G42.

Tool diameter D must be set before G41-G42 and it must not changed until the compensation remains active.

18.1.4 G47 – Set tool entry mode

Syntax

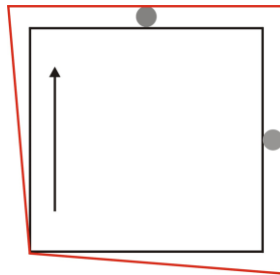
G47 Xval

Function type

MODAL

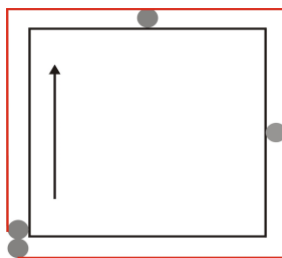
Description

G47 allow to choose two modes of tool entry. The default mode is **G47 X0**: compensation starts on next segment. With **G47 X1** compensation starts immediately and it is useful with closet shape.



Ex. Working with **G47 X0** (default)

Ex. Working with **G47 X1**



NOTE:

Using G47X1 tool must be positioned in start point byl PartProgram (so already shifted from shape of tool radium). The end of compensation instead must follow this scheme:

G0 Z (UP) // raise Z

G40 // disable compensation

G1 Z (UP) // raise Z again

Example for compensation of a square:

G47 X1

F 2.5

G0 X 26.2332 Y 26.1708 // move tool on start point

F 5

G1 Z 0 // move Z to work position

D 3 // set tool diameter

G42 // enable compensation G42

G1 X 24.7332 Y 26.1708

G1 X 24.7332 Y 76.1708

G1 X 74.7332 Y 76.1708

G1 X 74.7332 Y 26.1708

G1 X 24.7332 Y 26.1708

G0 Z 20 // raise Z

G40 // disable compensation

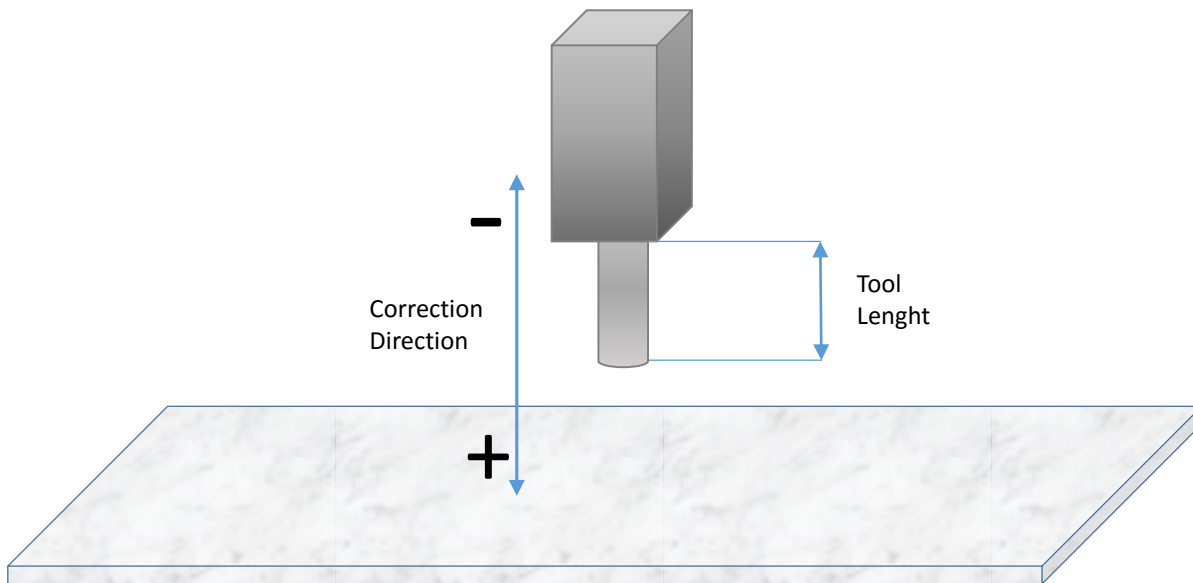
G1 Z 20 // raise Z again

18.2 TOOL LENGHT COMPENSATION

It's possible to compensate tool length on any axis. It is useful when we want to work the same PartProgram with tools of different length. Tool length compensation acts only in the direction selected by G43 function.

The axis affected by compensation is selectable as well as length of correction. Length correction compensates movement on the selected axis.

The effective tool length must be the measure of projection tools from spindle.



18.2.1 G43 – Enable tool length compensation from parameter

Syntax

G43 **X**len **K**mode **AXIS**dir(+/-) (DX Relative value)

Function type

MODAL

Cancel

G45 (optional correction set by Tool Table)

Parameters

len tool length in mm es: 12.35

If LEN=0 tool length is taken by tool table Tn

Mode Optional K parameter, this can have the following values:

K0 Normal Mode how if K is omitte

K1 With this parameter, the tool length is used how additional OFFSET , therefore the axis value (ABSOLUTE,RELATIVE) is immediatly updated
In this mode the "dir" parameter is not considered, the direction is defined by "LEN" sign (+ or -)

AXIS Axis on which apply length compensation

dir Direction of compensation

+ positive

- negative

Description

G43 enables tool length compensation. Corrections are applied only in the selected axis.

Ex:

G43 X10.3 Z+ // ENABLE POSITIVE COMPENSATION ON AXIS Z OF LEN=10.3mm

18.2.2 G44 – Disable tool length compensation G43

Syntax

G44

Function type

MODAL

Cancel

G43

Description

G44 disable tool length compensation set with G43

18.2.3 G45 – Enable tool length compensation from tool table T

Syntax

G45 AXISdir(+/-)

Function type

MODAL

Cancel

G43 (optional correction set by Tool Table)

Parameters

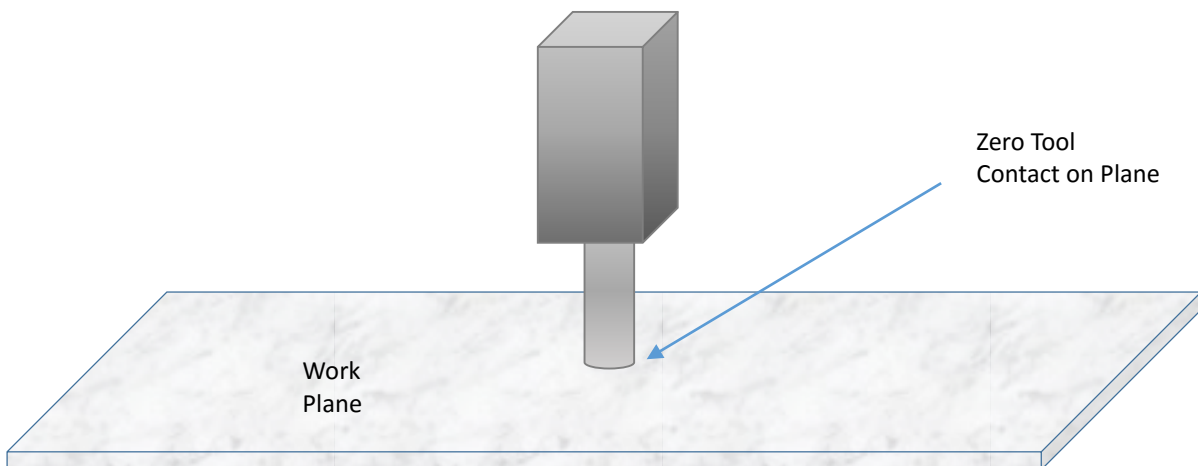
AXIS	Axis on which apply length compensation
dir	Direction of compensation
	+ positive
	- negative

Description

G45 works like G43 but the length of tool is taken only in the tool table with **Tn**.

Ex:

G45 Z+ // ENABLE POSITIVE TOOL LENGTH COMPENSATION ON AXIS Z



18.2.4 G46 – – Disable tool length compensation G45

Syntax

G46

Function type

MODAL

Cancel

G45

Description

G46 disable tool length compensation set with G45

18.3 INTERPOLATION MODES

IsoNs can use several axis interpolation modes allowing to adapt it in a lot of types of machine.

18.3.1 G60 – Enable FAST interpolation without stop on segments

Syntax

G60

Function type

MODAL

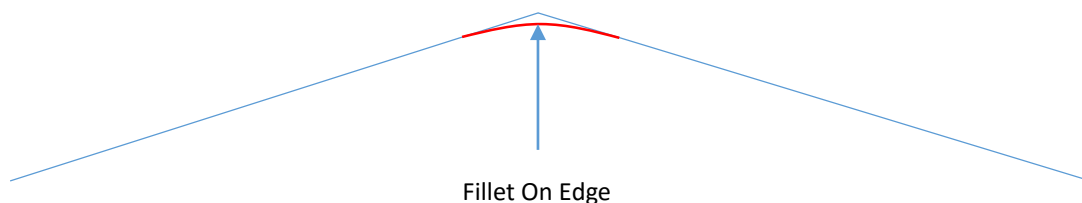
Cancel

G61

Description

G60 enables interpolation without stop on segments. The threshold of angle under which axis don't stop is set by machine parameter.

In the points where axis don't stop, it is generated an optimal trajectory minimizing error. **G60** avails a buffer of movements inside CN which can be set from a value up to **1024** on NG35 series. It allows to work in fluid way paths formed by micro-segments.



18.3.2 G61 – Enable interpolation with stop on segments

Syntax

G61

Function type

MODAL

Cancel

G60

Description

G61 enables interpolation with stop on segment. In this interpolation mode axis are stop at the end of each segment.

18.3.3 G62 – Wait for stop axis

Syntax

G62

Function type

MODAL

Description

G62 waits until axis are stopped and movement buffer is empty. It is useful when CN is in fast interpolation and we need to wait all movements are terminated. Obviously It is obsolete working in G61 (stop on segment).

18.3.4 G63 – Interpolation outside work plane (PX_MOVETO)

Syntax

G63

Function type

MODAL

Cancel

G64-G65

Description

G63 enables interpolation with axis outside work plane. In this way it's possible to work trajectories in the space (3D) without to consider the work plane.

The stop on segment is programmed by G62. G63 is particularly useful in 3D working where movement is formed by three or more axis. Generally in this interpolation mode are executed working from CAM at which is demanded the calculation of edge on segments inserting G62 in the fit point.

18.3.5 G64 – Interpolation on work plane (default)

Syntax

G64

Function type

MODAL

Cancel

G63-G65

Description

G64 enables interpolation mode considering the work plane. This is the default mode selected on CN. **G64** calculates vectorial speed only with the axis of work plane (PX_LINETO), moving consequently the other.

IsoNs verifies automatically if there are movement of the work plane axis, otherwise it works as G63.

Using this mode it can verify some inconvenient when the two axis of the work plane do very little movement. The following example explains that.

G68 MUST BE ENABLED

Ex:

G64

G17

G1X0.001Y0.001Z10

In the above example axis X and Y on work plane move very little in relation to axis Z causing an high speed on axis Z that can be bigger than its limit.

The solution to that is the following:

G63

G17

G1X0.001Y0.001Z10

G64

Enabling interpolation outside work plane, IsoNs calculates the vectorial speed with all axis instead only the two axis of work plane.

18.3.6 G65 – Enable 3D interpolation PX_MOVE TO with calculated stop on edge by parameters

Syntax

G65

Function type

MODAL

Cancel

G63-G64

Description

G65 Enables 3D interpolation calculating stop on edge by CN parameters.

G63-G64 MUST NOT BE ENABLED

Ex:

G65

G1X100Y100Z10

In the example PX_MOVE TO (interpolation outside work plane) is used and stop on edge is calculated automatically by CN.

18.3.7 G75 – Enable G64 for movement inside the work plane , G65 for movement outside the work plane

Syntax

G75

Function type

MODAL

Cancel

G63-G64

Description

G75 Enable interpolation 2D **G64** when the movements are inside the work plane with **SGLP**, Enable interpolation 3D **G65** when the movements are outside the work plane with **SGL3D**.

G63-G64 MUST NOT BE ENABLED

18.3.8 G66 – AFC – Adaptive Feed Control

Syntax

G66 Xval Yvin Zval Aval

Function type

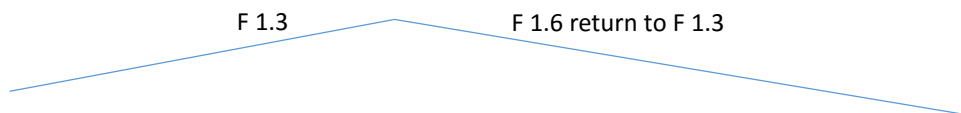
Disabled when PartProgram start

Parameters

Xval 0=disable AFC, 1=Enable AFC
if greater than 1, is carried out a moving average on the value of F calculated and the number of samples indicated in xVal. This makes it more "soft" means the change of the F calculated.

Yvin (optional) axis inertia
Vin > 1 deceleration space increases
If it isn't present Vin=1 is set (no inertia)

Zval (optional) Indicates the values of Delta F to below which are not considered.
ex:
Val = 0.3, a delta F of 0.3 or less is not considered and therefore the F is left to the previously calculated.



Aval Minimum value below which is not diminished the F
Ex: if Val = 0.1 to 0.1 lower values of F are limited to 0.1

ATTENTION

To correct working of G66, also G69 (look ahead depth) must be set

Description

G66 enables the **AFC** filter to auto-calculate interpolation speed by path type. With **SGLAFC_n** parameters, the **AFC** algorithm computes the optimal speed for each segment. The analysis of speed is carried out in the **look ahead** segments buffer (set by G69). Obviously more the buffer is deep, more the result is accurate. The **AFC** algorithm works only on **G1** segments and it makes sense only in paths formed by **MICRO-SEGMENTS**. It's basilar understand **AFC** can change the axis speed in each single segment and therefore the resulting speed can be inconstant along path. The **SGLAFC_** parameter of each single axis defines the instant speed gap it can bear without problems. If, in the analyzed segment, even a single axis exceeds its threshold, the speed is reduce to fit set parameters. In the presence of edges (defined by **SGL3D_n** parameters) axis are decelerated to a stop.

AFC is very useful in works where it doesn't need a constant speed along path. In such a cases **AFC** allows to move at speed as high as the path makes it possible, slowing in critical points (curves).

18.3.9 G67 – Px_Moveto for movement outside plane and Px_Lineto for inside one

Syntax

G67

Function type

MODAL

Cancel

G68

Description

G67 works with a combination of PX_MOVE TO for movement outside work plane (3D) and PX_LINETO for movements inside work plane. Furthermore when PX_MOVE TO is used axis are stop at end of segment.

G63-G65 MUST NOT BE ENABLED

G68

18.3.10 G68 – Always using of PX_LINETO in G1 “TRANSPORTED AXIS” (with possibility to combine with PX_MOVE TO)

Syntax

G68 Xval

Function type

MODAL

Cancel

G67

Parameter

Xval

Optional parameter (it must be enabled G69)

val>0 limit speed of eventually tangential axis

If tangential axis exceed this speed transported axis is disabled and PX_MOVE TO is invoked

Val=0 disable

Val=-1 axis is never transported, but it is only calculated the edge threshold on work plane (as G65 but with SGLP)

Description

G68 invokes always PX_LINETO for movement both inside and outside work plane, allowing to manage interpolation for particular machine types (for ex. cutting with tangent blade).

AXIS OUTSIDE WORK PLANE BECAME “TRANSPORTED”

These axis (outside plane), moving to final position with determination of vectorial speed by the only axis on work plane, don’t affect the resulting speed of tool.

An example of use with this interpolation mode can be the CUTTING PLOTTER one, where the blade must be tangent to path. Resulting working speed is really the programmed one by F, tangential axis instead will be moved to its final position in the same time of the other.

X parameter allows to obtain mixed interpolation between “TRANSPORTED AXIS” and normal axis.

Using this interpolation mode there can come out a problem: the “TRANSPORTED AXIS” can reach a speed over its limit. Setting this limit with X parameter IsoNs identifies such points and invoke PX_MOVE TO reducing resulting speed.

G63-G65 MUST NOT BE ENABLED

18.3.11 G69 – LHK – Buffer look ahead depth

Syntax

G69 Xval AXISTANG

Function type

Disabled when PartProgram start

Parameters

val *Number of elements in the buffer*
 If VAL<20 AFC is disabled
 VAL is limited to number of blocks of PartProgram

ASSIXTANG *Optional axis to be set as TANGENTIAL (Ex. Z A etc.)*
 If it isn't present no tangential axis is set

Description

G69 defines and enables the deep of **look ahead** segments buffer. It allows proper operation in functions which use it (G66 – G72 – G73 – G74). More the buffer is deep, more the result is accurate.

Depth is however limited to a value <20 (LHK disabled) to value major of part program length. A typical value can be 200.

This function also defines an optional tangential axis (for ex. cutting blade). This is necessary because functions related to **G69** (G66 – G72 – G73 – G74) can change the original path, adding or removing segments necessitating to recalculate the angle of tangent axis.

Ex:

G69 X200 A

In the above example buffer depth is set at 200 elements and axis A is set as TANGENTIAL.

18.4 ISONS FILTERS

IsoNs is equipped with several filtering algorithms to modify original path formed by PartProgram. Filter can be useful especially when PartPrograms comes from CAM (for ex. scanner acquisition) having a "NOISE" path.

Filters acts only in G1 blocks and however it must be enabled G69 function (LHK depth).

They can be **2D** features (work plane axis) or **3D** features (work plane axis and depth axis).

Filters however **DOESN'T ACT ON MORE THAN THREE AXIS**.

WARNING

The depth axis is automatically taken from axis XYZ which remains outside work plane.

Work plane X,Y Depth axis Z

Work plane X,Z Depth axis Y

Work plane Y,Z Depth axis X

When filtering is active current working lines in PART PROGRAM can be shifted to original ones.

It can happen some of these are removed and other are divided in more segments.

SEQUENCE OF FILTER

All available filters can be combined together, it can be invoked in following sequence:

NOISE	G73	(if enabled)
NURBS G72	(if enabled)	
FINE NOISE	G73	(if enabled)
RLS	G74	(if enabled)
SMOOTHING	G106	(if enabled)
AFC	G66	(if enabled)

18.4.1 G72 – N.U.R.B.S (Non Uniform Rational Bspline) (2D 3D)

Syntax

G72 Xval Yminlen Zorder Alenseg

Function type

Disabled when PartProgram start

Parameters

val	0 – disable NURBS filter 1 – enable NURBS filter
minlen	Minimum G1 segment length to be inserted in NURBS filter. Filter acts in a buffer which has a continuous series of segments with a length less than MINLEN parameter. When a segment greater than this length is met, the NURBS buffer is computer making an interpolation on points. The typical default value is 0.2 mm <u>Optional parameter, if it isn't present last value or default value is taken.</u>
order	Order of NURBS curve (value from 1 to 7) It defines the accuracy of resulting. Lower value corresponds to a best accuracy. order=1 original point are carried back. Typical default value is 3. <u>Optional parameter, if it isn't present last value or default value is taken.</u>
lenseg	It defines the minimum segments length at output curve resulting to points interpolation. At lower value corresponds to a less segments length and to a greater number of resulting segments (curve more accurate) Typical default value is 0.2mm <u>Optional parameter, if it isn't present last value or default value is taken.</u>

Description

NURBS are a type of curve suitable to solve B-Spline problem, mainly the impossibility to draw simple shape as circle. By this evolution it's possible to generate conic curves, and to represent very complex curves with less number of control points.

N.U.R.B.S. (Not Uniform Rational B-Spline) are rational curves defined by control points and its relative weights.

Increasing the weights value of a point the curve approaches to the point. On the contrary the point will cause a less variation on the curve.

ISONS uses this interpolation mode to "SOFTEN" a series of segments.

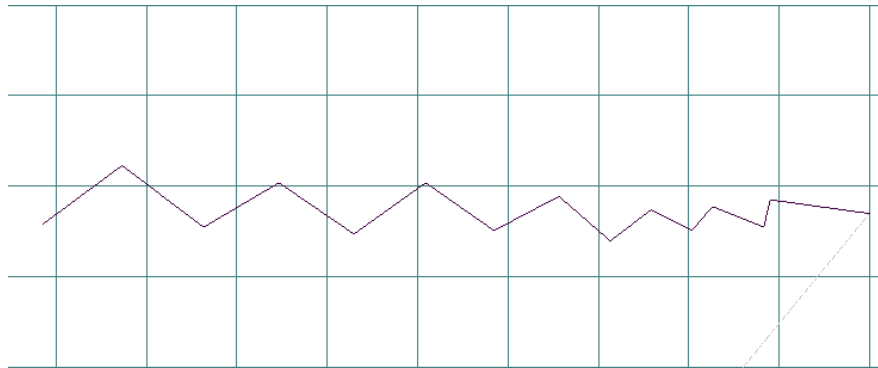
Some example are showed below to better explain NURBS filter.

NURBS filter acts also on 3D paths

NURBS FILTER IS THE SECOND TO BE INVOKED ON ORIGINAL PATH

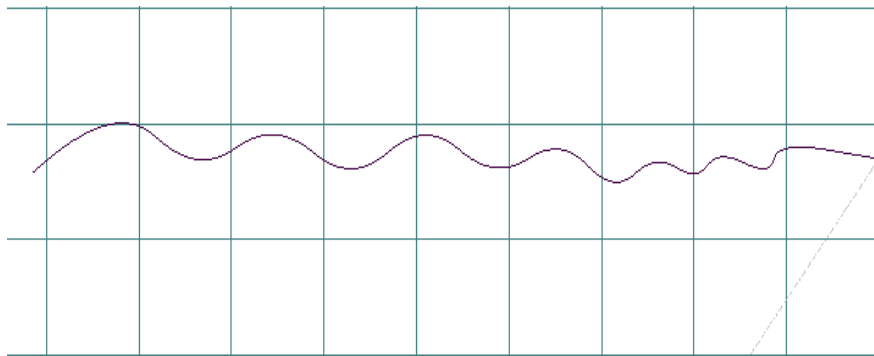
ORIGINAL PATH

```
G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59
```



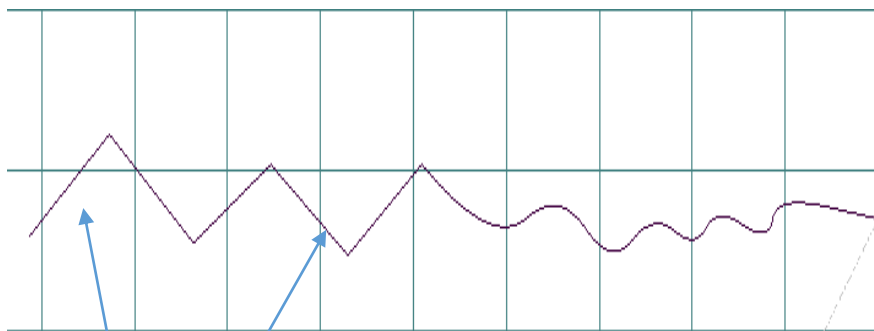
PATH FILTERED with NURBS MiLen=7mm Order=3 LenSeg=0.2mm

```
G69X200 // ENABLE LHK
// ENABLE NURBS LEN 7 MM ORDER 3 SEGMENT 0.2
G72X1Y7
G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59
```



PATH FILTERED with NURBS MiLen=5mm Order=3 LenSeg=0.2mm

```
G69X200 // ENABLE LHK
// ENABLE NURBS LEN 5 MM ORDER 3 SEGMENT 0.2
G72X1Y5
G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59
```



SEGMENT LENGTH >= 5 mm (not filtered)

PATH FILTERED with NURBS MiLen=7mm Order=5 LenSeg=0.2mm

With higher order NURBS path deviates most from the original path

G69X200 // ENABLE LHK

G72X1Y7Z5

G0 X107.16 Y130.27

G1 X100.59 Y131.18

G1 X100.14 Y129.37

G1 X96.74 Y130.73

G1 X95.38 Y129.14

G1 X92.66 Y130.5

G1 X89.95 Y128.46

G1 X86.55 Y131.41

G1 X82.24 Y129.14

G1 X77.71 Y132.31

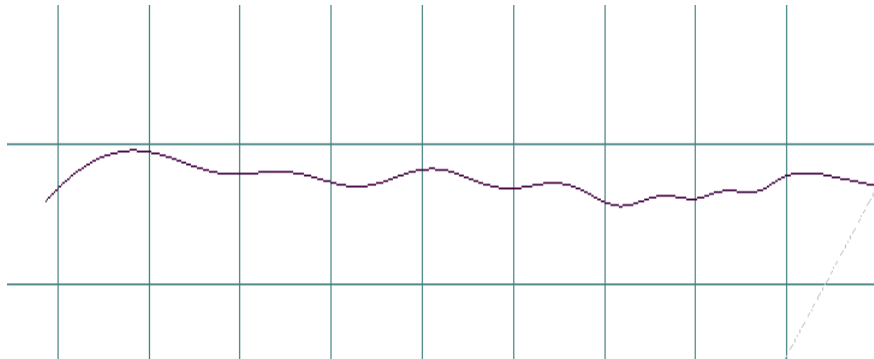
G1 X72.95 Y128.91

G1 X67.97 Y132.31

G1 X62.98 Y129.37

G1 X57.55 Y133.45

G1 X52.34 Y129.59



PATH FILTERED with NURBS MiLen=7mm Order=2 LenSeg=0.2mm

With Order=2 changes are almost null even in marked circles original path is slightly modified

G69X200 // ENABLE LHK

G72X1Y7Z1

G0 X107.16 Y130.27

G1 X100.59 Y131.18

G1 X100.14 Y129.37

G1 X96.74 Y130.73

G1 X95.38 Y129.14

G1 X92.66 Y130.5

G1 X89.95 Y128.46

G1 X86.55 Y131.41

G1 X82.24 Y129.14

G1 X77.71 Y132.31

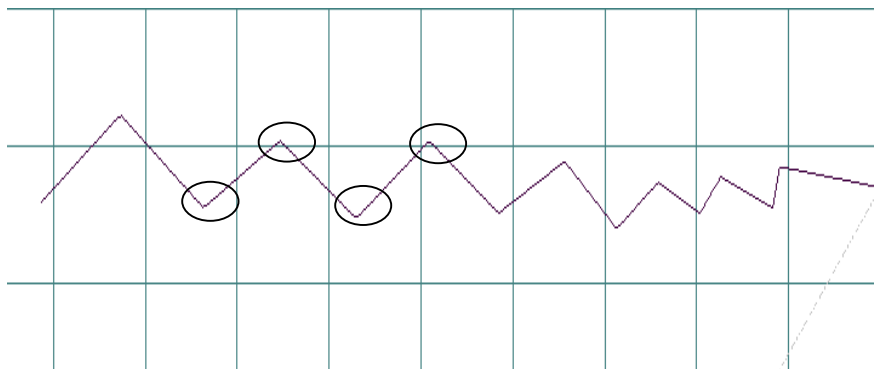
G1 X72.95 Y128.91

G1 X67.97 Y132.31

G1 X62.98 Y129.37

G1 X57.55 Y133.45

G1 X52.34 Y129.59



PATH FILTERED with NURBS MiLen=7mm Order=1 LenSeg=0.2mm

Original path not filtered

G69X200 // ENABLE LHK

G72X1Y7Z1

G0 X107.16 Y130.27

G1 X100.59 Y131.18

G1 X100.14 Y129.37

G1 X96.74 Y130.73

G1 X95.38 Y129.14

G1 X92.66 Y130.5

G1 X89.95 Y128.46

G1 X86.55 Y131.41

G1 X82.24 Y129.14

G1 X77.71 Y132.31

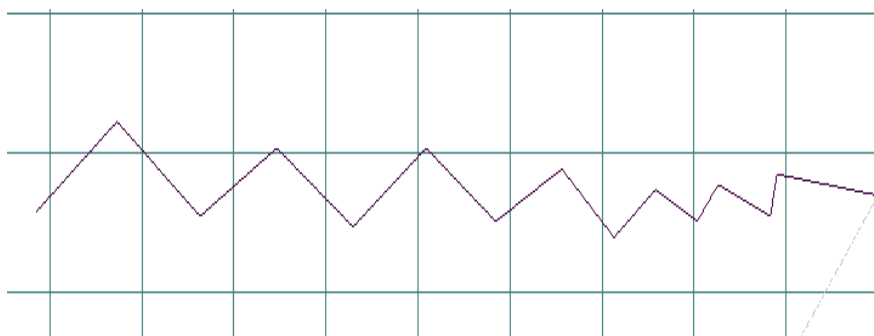
G1 X72.95 Y128.91

G1 X67.97 Y132.31

G1 X62.98 Y129.37

G1 X57.55 Y133.45

G1 X52.34 Y129.59



18.4.2 G73 – NOISE (2D)

Syntax

G73 Xval Yminang Zminlen Aonlyvar

Function type

Disabled when PartProgram start

Parameters

val 0 – disable NOISE, disable FINE NOISE
 1 – enable NOISE, disable FINE NOISE
 2 – disable NOISE, enable FINE NOISE
 3 – enable NOISE, enable FINE NOISE

minang Minimum edge angle to be consider noise

Typical default value is 25 degree

Optional parameter, if it isn't present last value or default value is taken.

minlen

Minimum G1 segment length to be inserted in NOISE filter.

Filter acts in a buffer which has a continuous series of segments with a length less than MINLEN parameter. When a segment greater than this length is met, the NOISE buffer is computer making an interpolation on points.

Typical default value is 0.4 mm

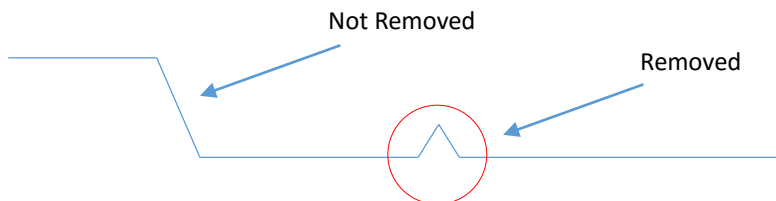
Optional parameter, if it isn't present last value or default value is taken.

onlyvar 0 – remove all edge matching the parameters

1 – remove only cusps matching the parameters (sudden changes of angle)

Typical default value is 0

Optional parameter, if it isn't present last value or default value is taken.



Description

NOISE filter helps to remove **G1** segments forming small edge (or cusps) which are defined as noise.

This is very useful in combination with NURBS filters increasing its performance.

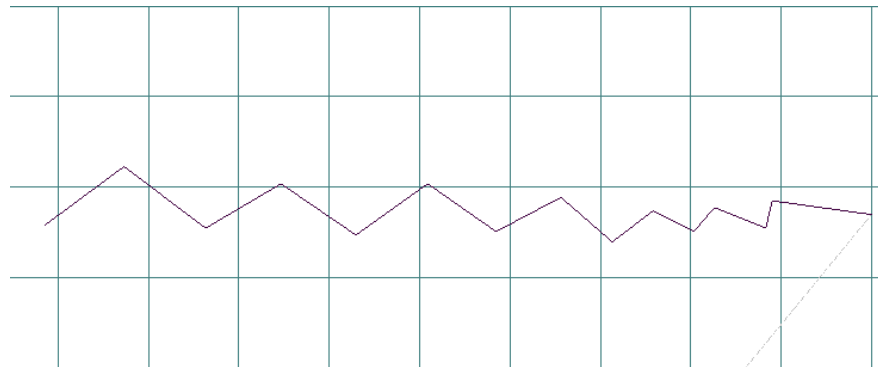
Some example are showed below to better explain NOISE filter.

NOISE filter acts only in 2D paths

NOISE FILTER IS THE FIRST FILTER IT IS INVOKED IN THE ORIGINAL PATH

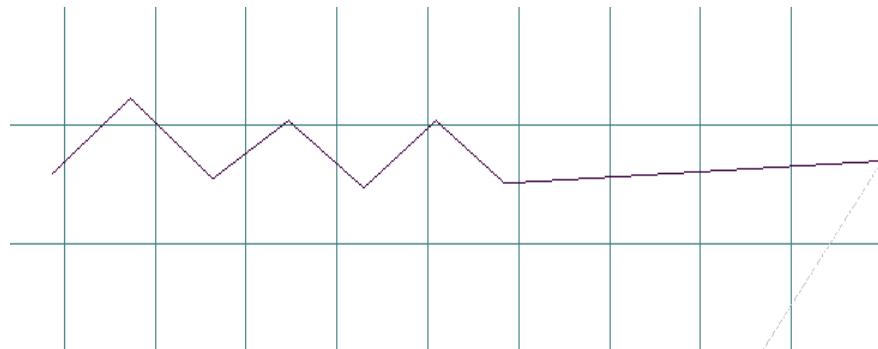
ORIGINAL PATH

G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59



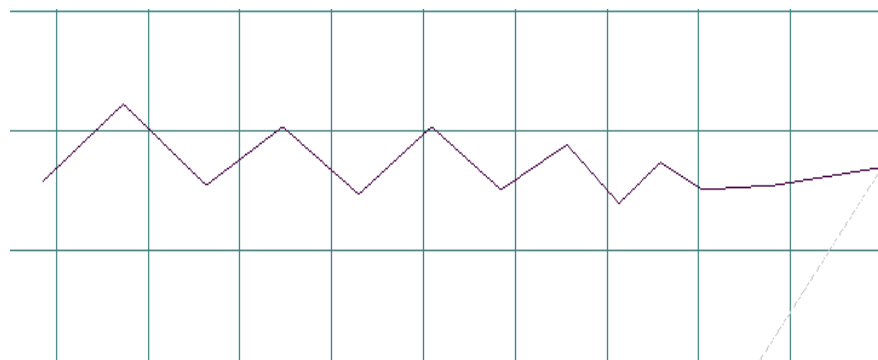
PATH FILTERED with MinAng=45 MinLen=5

G69X200 // ABILITA LHK
G73X1Y45Z5
G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59



PATH FILTERED with MinAng=45 MinLen=3

G69X200 // ABILITA LHK
G73X1Y45Z3
G0 X107.16 Y130.27
G1 X100.59 Y131.18
G1 X100.14 Y129.37
G1 X96.74 Y130.73
G1 X95.38 Y129.14
G1 X92.66 Y130.5
G1 X89.95 Y128.46
G1 X86.55 Y131.41
G1 X82.24 Y129.14
G1 X77.71 Y132.31
G1 X72.95 Y128.91
G1 X67.97 Y132.31
G1 X62.98 Y129.37
G1 X57.55 Y133.45
G1 X52.34 Y129.59



18.4.3 G74 – RLS Remove Len Segment (2D 3D)

Syntax

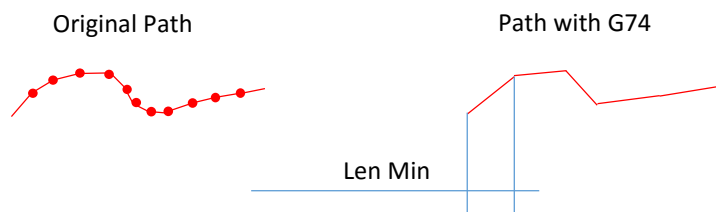
G74 Xval Ylenmin

Function type

Disabled when PartProgram start

Parameters

val	<i>0 – disable RLS filter 1 – enable RLS filter</i>
lenmin	<i>Minimum G1 segment length allowed. Segments with length less than this value are removed. Typical default value is 0.5 mm With lenmin=0 the value is automatically calculated by programmed speed (F). <u>Optional parameter, if it isn't present last value or default value is taken.</u></i>



Description

RLS filter allows to remove “**SHORT**” **G1** segments which couldn't be worked at requested speed (travel time less than CN sample). Setting a **ZERO** value, IsoNs calculates automatically the length optimizing it to current speed. The result is an exclusion of “void segment”, freeing the movement BUFFER and allowing greater work speed.

ATTENTION

When RLS is enable override potentiometer is disabled.

RLS filter atcs on both 2D and 3D paths

RLS FILTER IS 4th TO BE INVOKED ON ORIGINAL PATH

18.4.4 G106 – Smoothing (2D 3D)

Syntax

G106.ax Xval ALevel Bminlen

Function type

Disabled when PartProgram start

Parameters

.ax	Axis index where is set the filter (from 0 to n) normally 0 is X axis ex: G106.0
val	0 – Filter Smoothing disabled 1 - Filter Smoothing Enabled
Level	Smoothing level (from 0 to 1) Values low have more effect to smoothing ex. A0.3 Typical value is 0.7 <u>Optional parameter, if it isn't present last value or default value is taken.</u>
minlen	Minimum G1 segment length allowed. Segment more large this value, not are inserted in the filter Typical default value is disable The length is calculated by axes vector (for all axes). Is necessary set only one time If the parameter is not present, the last length value is used

Description

The **SMOOTHING** filter, is used to smoothing G1 segment, it allows to increase the execution speed, but decreased the precision respect to original path.

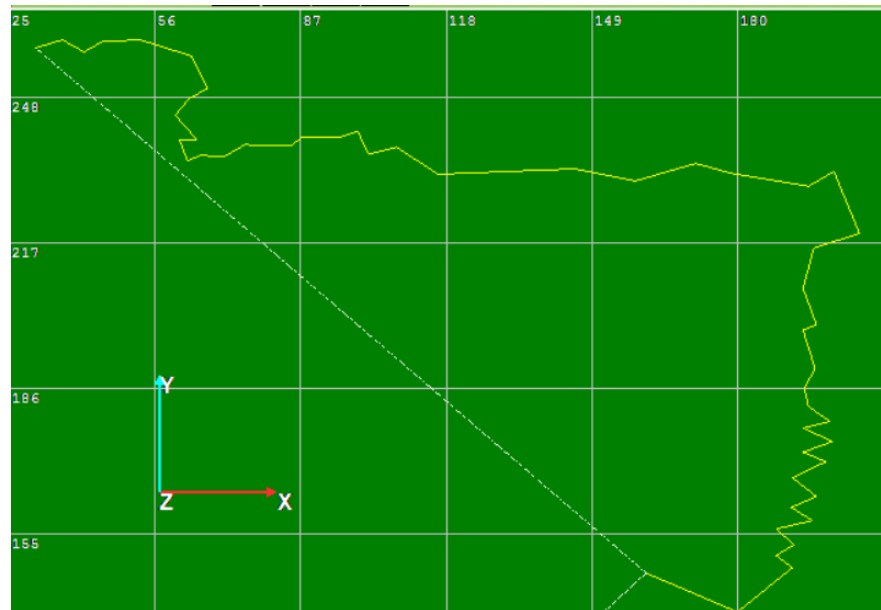
The **SMOOTHING** is activated for any axes with separate parameters.

THE SMOOTHING FILTER IS 5th TO BE INVOKED ON ORIGINAL PATH

ORIGINAL PATH

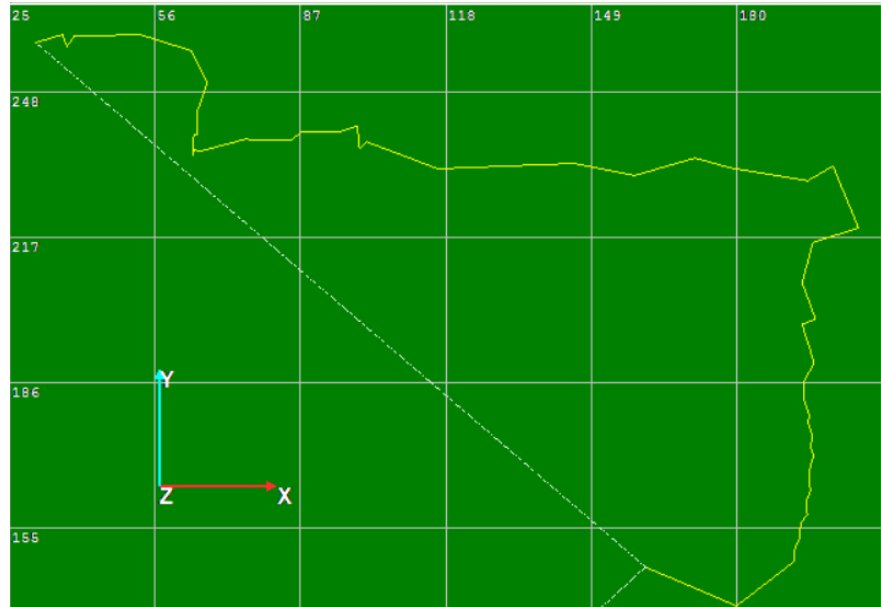
```

G60
G0 Z3
G0X30.4845 Y258.703
F15
G01 Z-1
X36.1945 Y260.416 F15
X40.7625 Y257.8465
X44.7595 Y260.1305
X52.7535 Y260.416
X63.6025 Y256.99
X67.0285 Y250.138
X63.0315 Y247.854
X60.1765 Y244.428
X64.7445 Y239.289
X61.033 Y239.289
X62.746 Y234.721
X65.601 Y235.863
X70.4545 Y235.5775
X75.0225 Y238.147
X84.7295 Y237.8615
X87.299 Y239.86
X95.5785 Y239.86
X99.0045 Y241.002
X101.2885 Y236.1485
X107.284 Y237.576
X116.1345 Y231.866
X145.2555 Y233.008
X158.103 Y230.4385
X170.9505 Y234.15
X178.088 Y232.1515
X194.9325 Y229.2965
X200.357 Y232.437
X205.7815 Y219.304
X196.0745 Y216.1635
X193.7905 Y207.5985
X196.6455 Y199.89
X193.7905 Y198.748
X196.36 Y190.4685
X194.076 Y186.4715
X194.9325 Y182.4745
X199.5005 Y179.334
X193.7905 Y177.9065
X200.0715 Y175.0515
X193.7905 Y172.7675
X198.644 Y170.769
X191.5065 Y167.343
X196.6455 Y163.346
X191.221 Y161.062
X195.789 Y158.207
X188.0805 Y156.494
X191.792 Y153.068
X188.0805 Y150.784
X191.5065 Y148.2145
X179.801 Y138.793
X160.387 Y147.0725
G0 X160.387 Y147.0725
    
```



PATH – Smoothing on X,Y level A0.2 Min Len Segment 8 mm B8

G60
G0 Z3
G69X200
G106.0 A0.2 B8
G106.1
G0X30.4845 Y258.703
F15
G01 Z-1
X36.1945 Y260.416 F15
X40.7625 Y257.8465
X44.7595 Y260.1305
X52.7535 Y260.416
X63.6025 Y256.99
X67.0285 Y250.138
X63.0315 Y247.854
X60.1765 Y244.428
X64.7445 Y239.289
X61.033 Y239.289
X62.746 Y234.721
X65.601 Y235.863
X70.4545 Y235.5775
X75.0225 Y238.147
X84.7295 Y237.8615
X87.299 Y239.86
X95.5785 Y239.86
X99.0045 Y241.002
X101.2885 Y236.1485
X107.284 Y237.576
X116.1345 Y231.866
X145.2555 Y233.008
X158.103 Y230.4385
X170.9505 Y234.15
X178.088 Y232.1515
X194.9325 Y229.2965
X200.357 Y232.437
X205.7815 Y219.304
X196.0745 Y216.1635
X193.7905 Y207.5985
X196.6455 Y199.89
X193.7905 Y198.748
X196.36 Y190.4685
X194.076 Y186.4715
X194.9325 Y182.4745
X199.5005 Y179.334
X193.7905 Y177.9065
X200.0715 Y175.0515
X193.7905 Y172.7675
X198.644 Y170.769
X191.5065 Y167.343
X196.6455 Y163.346
X191.221 Y161.062
X195.789 Y158.207
X188.0805 Y156.494
X191.792 Y153.068
X188.0805 Y150.784
X191.5065 Y148.2145
X179.801 Y138.793
X160.387 Y147.0725
G0 X160.387 Y147.0725



18.5 VIRTUAL AXIS

Virtual axis are commands inserted in the interpolation buffer but they don't refer to a real axis. In detail they are synchronous commando which can enable events in precise point of path. This commands must be used when CN works with fast interpolation. Furthermore command function must be implemented in CN.

18.5.1 G100 – Comando sincrono per asse virtuale

Syntax

G100 Xcmd

Function type

Direct

Parameter

Cmd *Command code to send to CN*

Description

G100 sends a command to CN synchronously with path. Such a command must be implemented in CN by VTB environment to work properly (refer to CN documentation).

Example to manage a digital output in fast interpolation.

PartProgram

G60 *//ENABLE FAST INTERPOLATION*

G1X100Y100

X120Y110

X130

G100 X1*// SET OUTPUT*

X200Y160

X210Y180

G100 X0*// CLEAR OUTPUT*

X300Y200

X310Y220

Two function must be implemented in CN for:

G100 X0 Clear output

G100 X1 Set output

This is the VTB code to be written in TaskPlc using output n.0

' G100 command for set/clear output

ISOV1.out0=INTERPOLA1.cmd

18.6 OTHER GENERIC G FUNCTIONS

18.6.1 G4 – Timed pause

Syntax

G4 Fvalue

Function type

Direct

Parameter

Value Time seconds (resolution up to tenth of second)

Description

G4 allows to pause PartProgram for the time set by F parameter.

Ex:

G4F2.5 // PAUSE OF 2.5 SEC

18.6.2 G10 – Enable external OVERRIDE potentiometer

Syntax

G10

Function type

Modal - default

Cancel

G11

Description

G10 enables speed control by external potentiometer.

18.6.3 G11 – Disable external OVERRIDE potentiometer

Syntax

G11

Function type

Modal

Cancel

G10

Description

G11 disables speed control by external potentiometer.

18.6.4 G101 – Axis Stop command

Syntax

G101

Function type

Direct

Description

G101 commands a stop to axis in current position. The example below explains this function.

Ex:

In this example axis X move to position 100. During positioning a digital input is tested (Input 0), if it activates the movements is stopped. If axis terminates its movement PartProgram is ended with an error.

```
G60
F1
G1X100
@INITLOOP
IF $[Q0]=100    // IF X REACHES 100
    GOTO ERR    // ERROR
END_IF
IF $[I0]=1      // WAIT FOR INPUT
    G101        // IF ACTIVE, AXIS STOP AND EXIT
    GOTO EXIT
ELSE
    GOTO INITLOOP // CONTINUE LOOP
END_IF
@EXIT
.
.

@ERR
.
.
```

18.6.5 G80 – Forced pause by code

Syntax

G80 Xcode

Function type

Direct

Parameter

Code *Code of pause (refer to configuration)*

Description

G80 forced CN to pause state and setting a pause code. **CODE** parameter defines the message code to be showed. If code hasn't been configured a message "NO CODE PAUSE" is showed. The PAUSE procedure is the same of the hardware one.

Ex:

G80 X4 // PAUSE IS FORCED WITH CODE 4

18.6.6 G81 – Management secondary axes LIMITS

Syntax

G81 Xmode

Function type

Direct

Parameter

mode

Mode of LIMITS management

X0 Set FIRST **POSITIVE** LIMITS (default – LIMITE_P_)

X1 Set FIRST **NEGATIVE** LIMITS (default – LIMITE_N_)

X2 Set SECOND **POSITIVE** LIMITS (2ND_LIMITE_P_)

X3 Set SECOND **NEGATIVE** LIMITS (2ND_LIMITE_N_)

Description

Sometimes is necessary change in temporary mode the Axes limits for make the axes movements outside to primary limits (Eg: TOOL CHANGE)

In IsoNs is possible used the SECONDARY LIMITS positive and negative

Machine parametrs:

2ND_LIMITE_N_

2ND_LIMITE_P_

This parameters can be activated or disactivated by G81 function.

If the secondary limits remain activated, when the PART PROGRAM is finished, automatically are reset to primary LIMITS.

This function is typically used to TOOL CHANGE MACRO (M6)

Ex:

G81 X3 // ACTIVE SECONDARY NEGATIVE LIMITS

.

G81 X1 // ACTIVE PRIMARY NEGATIVE LIMITS

18.6.7 G20 – Axes Values in Inch

Syntax

G20

Function type

Modal

Cancel

G21

Description

G20 Enable the Programmation Axes Values in Inch.(G0-G1-G2-G3 etc)

18.6.8 G21 – Axes Values in Millimeters

Syntax

G21

Function type

Modal - Default

Cancel

G20

Description

G20 Enable the Programmation Axes Values in Millimeters.(G0-G1-G2-G3 etc)

18.7 VARIABLE MANAGEMENT FUNCTIONS

IsoNs manages files which can contain the state of some variable. These files are stored on the PC HardDisk. It's possible to load them (one at a time) by Partprogram. When file has been loaded a list of saved variables is filled. The length of the list depends to the number of stored variable. There is no limit to list length. These variables can be read or written in the list by specific instructions in the PartProgram. In other words it's possible to obtain Backups of variables state for generic use.

18.7.1 LOAD_VAR – load a variables file in current list

Syntax

LOAD_VAR filename

Parameter

Filename	Name of the file to be loaded
	<i>The name can contain extension but there must not be delimitation characters (+-* etc.)</i>

Description

LOAD_VAR load the file named initializing the internal list with values container in the file. The list is defined as an ARRAY of double values and we can access them by **GET_VAR** and **WRITE_VAR** functions. If the file will be not found IsoNs will show the relative error.

Ex:

```
LOAD_VAR MYFILE.PNT // LOAD FILE
GET_VAR $VAR1 0 // SET VAR1 FROM INDEX 0 OF THE LIST
GET_VAR $VAR2 1 // SET VAR2 FROM INDEX 1 OF THE LIST
```

18.7.2 GET_VAR – Read a value from the loaded list and store it in a variable

Syntax

GET_VAR \$var index

Parameters

\$var	Name of the destination variable
Index	Expression indicating the index in the list

Description

GET_VAR reads a value in the current list and stores it in a variable. The list must have been sized (by DIM_VAR) or however it must have a size greater or equal to INDEX. If index is over the list size an error will be showed.

Ex:

```
LOAD_VAR MYFILE.PNT // LOAD FILE
$INDEX=0
LOOP 10
    GET_VAR $VAR1 $INDEX // LOAD X POSITION
    GET_VAR $VAR2 $INDEX+1 // LOAD Y POSITION
    G1 X[$VAR1]Y[$VAR2] // MOVE
    $INDEX=$INDEX+2 // INCREASE INDEX
END_LOOP
```

18.7.3 WRITE_VAR – Write a value in the loaded list getting it from a variable

Syntax

WRITE_VAR \$var index

Parameters

<i>\$var</i>	<i>Name of the source variable</i>
<i>index</i>	<i>Expression indicating the index in the list</i>

ATTENTION: list must be sized for INDEX value

Description

WRITE_VAR allows to store a value of the current list in a variable. The list must have been sized (by DIM_VAR) or however it must have a size greater or equal to INDEX. If index is over the list size an error will be showed.

Ex:

```
DIM_VAR 10           // SIZE LIST WITH 10 EMPTY ELEMENTS
WRITE_VAR $VAR1 0     // STORE VAR1 IN INDEX 0 OF THE LIST
WRITE_VAR $VAR1 1     // STORE VAR2 IN INDEX 1 OF THE LIST
SAVE_VAR MYFILE.PNT  // SAVE FILE
```

18.7.4 SAVE_VAR – save a variables file with current list

Syntax

SAVE_VAR filename

Parameter

<i>filename</i>	<i>Name of the file to be saved</i>
-----------------	-------------------------------------

The name can contain extension but there must not be delimitation characters (+-* etc.)

Description

SAVE_VAR saves the value in the current list to the named file.

Ex:

```
DIM_VAR 10           // SIZE LIST WITH 10 EMPTY ELEMENTS
WRITE_VAR $VAR1 0     // STORE VAR1 IN INDEX 0 OF THE LIST
WRITE_VAR $VAR1 1     // STORE VAR2 IN INDEX 1 OF THE LIST
SAVE_VAR MYFILE.PNT  // SAVE FILE
```

18.7.5 FILE_EXISTS – test if a file exists

Syntax

FILE_EXISTS \$var filename

Parameters

<i>\$var</i>	<i>Variable where to store result:</i> <i>0=file not found, 1=file exists</i>
<i>filename</i>	<i>Name of the file</i> <i>The name can contain extension but there must not be delimitation characters (+-* etc.)</i>

Description

FILE_EXISTS allows to test if a variable file exists.

Ex:

```
FILE_EXISTS $VAR MYFILE.PNT // TEST FILE
IF $VAR=0                  // FILE NOT FOUND
....
END_IF
```

18.7.6 ADD_VAR – Add a value to current list

Syntax

ADD_VAR value

Parameter

value *Expression or variable which value will be appended to the list*

Description

ADD_VAR appends a value to the current list. The list will be resizing to contain the new element.

Ex:

```
ADD_VAR $VAR1           // APPEND VALUE FROM VAR1
ADD_VAR 10.23           // APPEND VALUE FROM CONSTANT
ADD_VAR $VAR2*($VAR3+$VAR4) // APPEND VALUE FROM EXPRESSION
```

18.7.7 REMOVE_VAR – Remove a value from current list

Syntax

REMOVE_VAR Index

Parameter

index *Expression indicating index of list to be removed*

Description

REMOVE_VAR remove a value from the current list. All index above the removed one will be re-indexed. If the index is over the list size an error will be showed.

Ex:

```
REMOVE_VAR 1 // REMOVE VALUE AT INDEX 1
```

18.7.8 CLEAR_VAR – Remove all values from the current list

Syntax

CLEAR_VAR

Description

CLEAR_VAR erase all the values from the current list. List will be sized at zero and the value cannot be restored.

Ex:

```
CLEAR_VAR           // CLEAR ALL CURRENT LIST
```

18.7.9 DIM_VAR – Size current list to a specific number of elements

Syntax

DIM_VAR Nelem

Parameter

<i>Nelem</i>	<i>Expression indicating the number of elements to be sized (the value of all elements will be set with ZERO)</i>
--------------	---

Description

DIM_VAR allows to re-size the current list. All last value will be lost and the new values will be set with ZERO.

Ex:

```
DIM_VAR 10           // SIZE LIST WITH 10 ELEMENTS
```

18.7.10 COUNT_VAR – Get the number of elements in the current list

Syntax

COUNT_VAR \$var

Parameter

<i>\$var</i>	<i>Variable where to store the list size</i>
--------------	--

Description

COUNT_VAR gets the number of elements of the current list and store the value in a variable.

Ex:

```
COUNT_VAR $VAR1      // STORE SIZE IN VAR1
IF $VAR1=0           // IF LIST IS EMPTY
    DIM_VAR 10        // RE-SIZE THE LIST
```

18.8 HM FUNCTIONS

HM functions are an upgrade to standard M functions. They can use parameters passed to function when invoked like common high level languages as C.

Parameters are then read by HM function writing it in private variables.

HM functions reside on PC and they must be created with the specific utility of human interface.

18.8.1 Call of HM functions

Syntax

HMnum par1 par2 par3

Function type

Direct

Parameters

Num *Number code of HM function*

PARn *Variable or constant. Parameters must be separated by space*

Description

HM with code **NUM** is invoked passing it the parameters defined with **PARn**.

HM functions always wait for emptying of the MOVEMENT BUFFER before it is really invoked.

All parameters defined in the declaration must be inserted, else an error will be generated.

Ex:

HM10 \$VAR1 \$VAR2 12.3 24.125 *// CALL OF HM 10 PASSING 4 PARAMETERS*

18.8.2 Building of a HM function

A **HM** function, before to be used, must be built and pre-compiled by specific option in the human interface.

HM functions are formed by ISO code and they can be used all IsoNs functions. After pre-compiling **HM** is available to PartProgram.

VARIABLES and **LABELS** declared inside HM function are **PRIVATE** (not visible in PartProgram). In other words

VARIABLES and **LABEL** could be a same name both in HM function and PartProgram, IsoNs recognized that automatically. It's possible to share **VARIABLES** with Partprogram declaring them with **GLOBAL** prefix.

Inside **HM** function it's possible call other **HM** or **M** functions both on CN and PC.

Example of building of a HM function

```
GET $PAR1 $PAR2 $PAR3 $PAR4 // READ PARAMETERS
GLOBAL $PAUSE                // SHARE $PAUSE WITH PART PROGRAM
F$PAR3                        // SET FEED AT $PAR3
G1 X$PAR1 Y$PAR2 Z$PAR3      // LINEAR INTERPOLATION
G4 F$PAUSE                    // PAUSE
```

As we can see from the example, a HM function is very similar to the high level languages. If there was a need to return a value we can do it using **GLOBAL** variables.

18.9 M FUNCTIONS

There are two type of M function:

- **Inside CN**
- **Inside PC**

Obviously function cannot have the same number. **M inside PC** are priority to **M inside CN**. If both they were present IsoNs invokes the PC one.

18.9.1 M functions inside CN

M function inside CN have to be built with VTB environment (refer to relative documentation). They can read some parameters from PartProgram or return it a value by some predefined variables.

PREDEFINED VARIABLES

<code>\$_PARAM_1</code>	parameter 1
<code>\$_PARAM_2</code>	parameter 2
<code>\$_PARAM_3</code>	parameter 3
.	
.	
<code>\$_PARAM_n</code>	parameter n

The number of parameters available depends to IsoNs configuration, but it's a limit of 10 parameters (default 5). Each time PartProgram writes or reads a variable of `$_PARAM_n` type, they are read from CN memory. `$_PARAM` variables are 32 bit **INTEGER**. All parameters must be written before invoking M function and read at the end of it. **M functions always wait for emptying of the MOVEMENT BUFFER before it is really invoked.**

Ex:

```
$_PARAM_1=134 // WRITE PARAMETER 1
$_PARAM_2=$VAR// WRITE PARAMETER 2
$_PARAM_3=12600 // WRITE PARAMETER 3
M100 // INVOKE M100 FUNCTION
$VAR1=$_PARAM_1 // READ PARAMETER 1 MODIFIED BY M100
```

18.9.2 M function inside PC

M function inside PC are very similar to HM functions, the only difference is that M functions cannot have any parameter. Therefore exchange of value must be made with GLOBAL variables.

A **HM** function, before to be used, must be built and pre-compiled by specific option in the human interface.

HM functions are formed by ISO code and they can be used all IsoNs functions. After pre-compiling **HM** is available to PartProgram.

VARIABLES and **LABELS** declared inside HM function are **PRIVATE** (not visible in PartProgram). In other words

VARIABLES and **LABEL** could be a same name both in HM function and PartProgram, IsoNs recognized that automatically. It's possible to share VARIABLES with Partprogram declaring them with **GLOBAL prefix**.

Inside **HM** function it's possible call other **HM** or **M** functions both on CN and PC.

Example of building of a M function

```
GLOBAL $VAR1           // SHARE $VAR1 WITH PART PROGRAM
GLOBAL $VAR2           // SHARE $VAR2 WITH PART PROGRAM
G1 X$VAR1 Y$VAR2       // LINEAR INTERPOLATION
```

18.10 Essential M FUNCTIONS

In this section are described the essential M functions. These are necessary for correct machine operation.

18.10.1 START M

This function is execute when START PROGRAM is required. Its use is not important, because the preparatory M functions are inserted in to Part Program

18.10.2 END M

This function is execute when END PROGRAM is required. Its use is not important, because the preparatory M functions are inserted in to Part Program.

18.10.3 STOP M

This function is execute when STOP PROGRAM button is pressed. This function is VERY IMPORTANT. It usually STOP ALL AXES and Spindle, water etc

Essential Functions In M STOP

- 1) **G101 Stop axes**
- 2) **G62 Wait axes stop**
- 3) **SET/RESET output for spindle,water etc.**
- 4) **Eventually parking axes**

Se nessuna M di STOP viene configurata, il CNC blocca solamente gli assi nel punto di STOP.
If no M STOP configured, will stop only the axes by CNC.

18.10.4 PAUSE M

This function is execute when PAUSE PROGRAM button is pressed. This function is VERY IMPORTANT. usually this function stops the spindle. the water and saves the positions axes for pause resume.

Essential Functions In M PAUSE (ex 3 axes)

- 1) **GLOBAL \$SAVEX GLOBAL variable for X AXIS**
- 2) **GLOBAL \$SAVEY GLOBAL variable for Y AXIS**
- 3) **GLOBAL \$SAVEZ GLOBAL variable for Z AXIS**
- 4) **G62 Wait axes stop**
- 5) **\$SAVEX=\$[Q0] Save X position**
- 6) **\$SAVEY=\$[Q1] Save Y position**
- 7) **\$SAVEZ=\$[Q2] Save Z position**
- 8) **SET/RESET output for spindle,water etc.**
- 9) **Eventually parking axes**

If M PAUSE is not configured, will stop only the axes by CNC.

18.10.5 RESUME PAUSE M

This function is execute when START button is pressed after PAUSE PROGRAM. This function is VERY IMPORTANT. usually this function repositions the axes and SET/RESET output for spendle, water etc.

Essential Functions In M RESUME PAUSE (ex 3 assi)

- 1) GLOBAL \$SAVEX GLOBAL variable for X AXIS
- 2) GLOBAL \$SAVEY GLOBAL variable for Y AXIS
- 3) GLOBAL \$SAVEZ GLOBAL variable for Z AXIS
- 4) G96 Offset suspension
- 5) G98 Work origins suspension
- 6) G0 Z0 Z axis to safety position
- 7) F1 SPEED axes
- 8) G1 X[\$SAVEX] Y[\$SAVEY] Axes X,Y, etc. To initial positions (these are saved by M pause)
- 9) G62 Wait axes in positions
- 10) G1 Z[\$SAVEZ] Axis Z to initial position (this is saved by M pause)
- 11) G62 Wait Z axis in position
- 12) G97 ResumeOffset
- 13) G99 Resume Work origins
- 14) SET/RESET output for spendle,water etc.

If no M resume pause configured, the CNC axes back to the starting point in the interpolation of a set F in the machine parameter "VRIPOS"

18.10.6 GO BLOCK M

This function is execute when START button is pressed by PlugIn "GO BLOCK". usually this function repositions the axes and SET/RESET output for spendle, water etc.

Essential Functions In GO BLOCK M (ex 3 assi)

- 1) \$POSX=\$[C0] Axis X position calculated by CNC
- 2) \$POSY=\$[C1] Axis Y position calculated by CNC
- 3) \$POSZ=\$[C2] Axis Z position calculated by CNC
- 4) G96 Offset suspension
- 5) G98 Work origins suspension
- 6) G0 Z[0] Z axis to safety position
- 7) F1 SPEED axes
- 8) G1 X[\$POSX] Y[\$POSY] Axes X,Y, etc. To initial positions
- 9) G62 Wait axes in positions
- 10) G1 Z[\$POSZ] Axis Z to initial position
- 11) G62 Wait Z axis in position
- 12) G97 ResumeOffset
- 13) G99 Resume Work origins
- 14) SET/RESET output for spendle,water etc.

18.10.7 GO RETRACE M

This function is execute when START button is pressed by PlugIn "RETRACE". usually this function repositions the axes and SET/RESET output for spendle, water etc.

Essential Functions In M RETRACE

- 1) SET/RESET output for spendle,water etc.

18.11 DEFINING DEPTH AXIS

The depth axis is used for some functions IsoNs. Normally, this axis is related to the work plan selected:

<i>Work Plane XY</i>	<i>Depth axis Z</i>
<i>Work Plane XZ</i>	<i>Depth axis Y</i>
<i>Work Plane YZ</i>	<i>Depth axis X</i>

The depth axis is used by the simulation process and the way PREVIEW and by G47 function

18.11.1 G48 Define Depth axis

Syntax

G48 axis name

Function type

immediate

Parameters

Axis Name *ex: X,Y,Z,A,B etc.*

Description

G48 definisce un asse di profondità diverso da quelli scelti in automatico da IsoNs.

Se l'asse scelto è uguale ad uno degli assi del piano di lavoro impostato, viene generato un errore di Run Time.

G48 defines a depth axis different to axis chose automatically by IsoNs

Ex:

G48 A // A is depth axis

18.12 MILD MODE – EDGE SMOOTHING

IsoNs can use a special algorithm for EDGE SMOOTHING

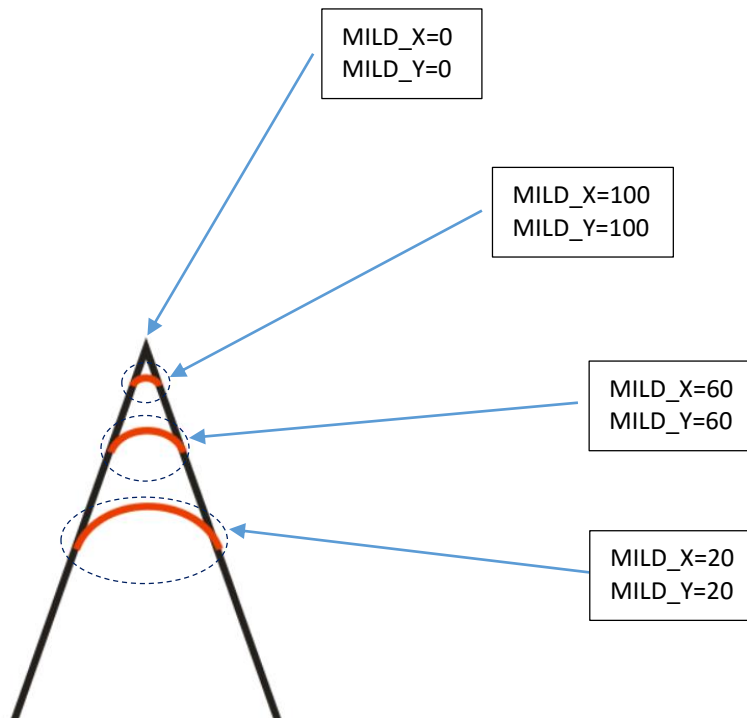
With the PARAMETERS MILD_ (MILD_X – MILD_Y etc.) you can Smooth the edge with various value.

The MILD MODE can be activated on single Axis.

The MILD_ set of value can be used from 200 to 20 (estimated values)

The G49 function, manages the MILD MODE.

Following you can see a MILD MODE example values



18.12.1 G49 MILD MODE Managing

Syntax

G49 Axes Name where enable MILD MODE – None Axis, MILD MODE DISABLED ON ALL AXES

G49.0 Suspends MILD MODE

G49.1 Resumes MILD MODE on the Axes indicated in previous G49

Function type

Immediate – Disabled on PROGRAM STOP

Parameters

Axes Name

Name of Axes where enable Mild Mode: ex: G49 XYZ etc..

Description

G49 manages ENABLE/DISABLE MILD MODE.

G49 with **Axes Name** (ex. G49 XY) enabled MILD MODE on the Axes indicated.

G49 Without parameters DISABLES MILD MODE on the ALL AXES.

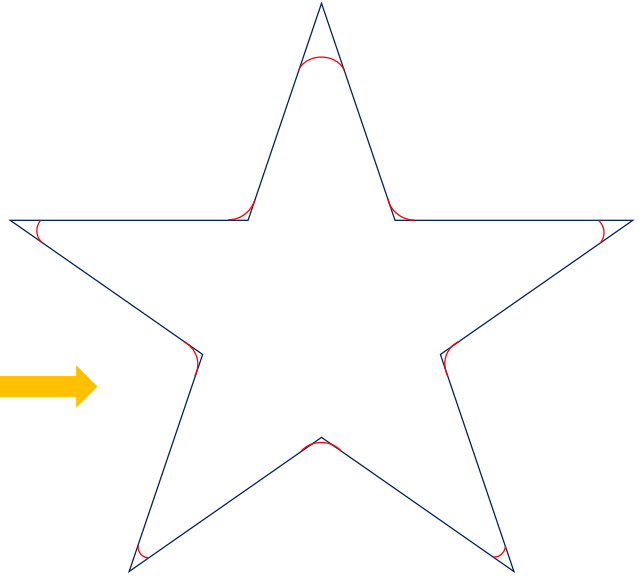
G49.0 Without parameters **SUSPENDS MILD MODE**

G49.1 Without parameters **RESUMES MILD MODE** if it was activated by **G49**.

Ex:

```

G60
F15
G0 Z3
G49 XY // ENABLED MILD MODE ON XY
G0X63.7738 Y160.6052
F10
G01 Z0
X70.9693 Y140.2459 F15
X92.5556 Y139.694
X75.4163 Y126.5593
X81.5619 Y105.8589
X63.7738 Y118.1006
X45.9857 Y105.8589
X52.1313 Y126.5593
X34.992 Y139.694
X56.5783 Y140.2459
X63.7738 Y160.6052
G49 // DISABLED MILD MODE ON ALL AXES
G0 Z3
G0 X56.5783 Y140.2459
    
```



18.13 MAIN MACHINE PARAMETERS

In this chapter are explained the MAIN machine parameters to be set in IsoNs to configure the machine. MAIN MACHINE PARAMETERS are always present on any application, however it's possible that applications share more PARAMETERS. Indeed IsoNs allows to customize parameters to adapt IsoNs for any type of machine.

NOTE TO CALCULATE ACCELERATION IN Mt/sec²

In IsoNs acceleration is expressed in increase for CN sample, to obtain the value in Mt/sec² this is the calculation to do:

$$ACC = ACC\_xx / TAU^2$$

TAU = Sample time set on CN in msec

ACC_xx = value of acceleration parameter

If for example we set an ACC_xx=100 and a sample of TAU=5 msec, we obtain:

$$ACC = 100 / (5 * 5) = 4 \text{ Mt/sec}^2$$

18.13.1 Generic parameters

18.13.1.1 FEEDMAX

It defines the maximum **FEED** of the system. The unit of measure depends by system resolution (typically mm/min). If it's set a **F** at speed greater than this, it is limited at **FEEDMAX**.

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typical 0.001) this parameter must be expressed in tenth of mm/min.

18.13.1.2 FEEDMIN

It defines the minimum **FEED** of the system. The unit of measure depends by system resolution (typically mm/min). If it's set a **F** at speed less than this, it is limited at **FEEDMIN**.

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typical 0.001) this parameter must be expressed in tenth of mm/min.

18.13.1.3 FEEDDEF

It defines the default **FEED** set by the system at start-up. The unit of measure depends by system resolution (typically mm/min).

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typical 0.001) this parameter must be expressed in tenth of mm/min.

18.13.1.4 FEEDRES

It defines the resolution of the speed set by F.

1	mm/min	ex: F1500 = 1,5 Mt/min
1000	mt/min	ex: F1.5 = 1.5 Mt/min

18.13.1.5 SPEEDMAX

It defines the maximum speed (**S**) of the system. The unit of measure depends by system resolution. If it's set a **S** at speed greater than this, it is limited at **SPEEDMAX**.

18.13.1.6 SPEEDMIN

It defines the minimum speed (**S**) of the system. The unit of measure depends by system resolution. If it's set a **S** at speed less than this, it is limited at **SPEEDMIN**.

18.13.1.7 SPEEDDEF

It defines the default speed (**S**) set by the system at start-up. The unit of measure depends by system resolution.

18.13.1.8 VMAXG0

It defines the maximum speed for the positioning with G0. It is in percent of the speed calculated by system based to maximum speed of each axis.

Value can be from 0 to 100 %.

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typycal 0.001) this parameter must be expressed in tenth of mm/min.

18.13.1.9 ACC_G0

It defines the acceleration for positioning with G0. See the above note to detail for unit of measure.

18.13.1.10 ACC_LAV

It defines the acceleration for interpolation with G1, G2 and G3. See the above note to detail for unit of measure.

18.13.1.11 ACC_QSTOP

It defines the acceleration for quick-stop when the position limit is reached. See the above note to detail for unit of measure.

18.13.1.12 VEL_GO_LINE_RETRACE

It defines the speed for positioning "GO LINE" during **RETRACE** function. The unit of measure depends by system resolution (typically mm/min).

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typycal 0.001) this parameter must be expressed in tenth of mm/min.

18.13.1.13 ACC_RAGGIO_MAX

It defines the centrifuge acceleration to auto-reducing speed on circular interpolation. It must be set after machine test. Normally this parameter is set with values from 1 to 15

Small values defines a greater feed reduction when the radius are small. If the minimum value 1 is not sufficient to reduction your desired feed, is necessary use the decimal part in the following mode (this function is enabled only in the ComSynk rev: 2.0.5.22 or greater):

Insert the parameter ACC_RAGGIO_MAX with value greater to 100, the effective value is:

$ACC\_RAGGIO\_MAX\_EFFECTIVE = (ACC\_RAGGIO\_MAX - 100) / 10$

So:

109 = 0,9

108 = 0,8

Etc.

18.13.1.14 VRIPOSO

It defines the speed for re-positioning in start after PAUSE. The unit of measure depends by system resolution (typically mm/min). This speed acts only if axis are moved in PAUSE state and they aren't repositioned in the exact point when restarted.

18.13.1.15 SGLP

Threshold in tenth of degree (default 200 = 20 degree) to stop axis on edge when CN works in FAST INTERPOLATION. Threshold acts only on work plane axis. CN stop automatically the axis when on work plane it recognizes an angle greater than SGLP (edge found).

18.13.1.16 SGLP_RED

Percentage of SGLP to begin the gradual slowdown in the proportion of the edge. If SGLP_RED = 0 parameter is disabled and is done only stop on the corner SGLP, otherwise the percentage set, the CNC starts slowing down (see examples below). When SGLP_RED is set to a value greater than 0, it would be necessary to increase the value of SGLP.

18.13.1.17 MAX_RED

Maximum percentage reduction in speed caused by the parameter SGLP_RED. This is always limited to a value below 100% (to avoid that the speed goes to ZERO)
(see examples below)

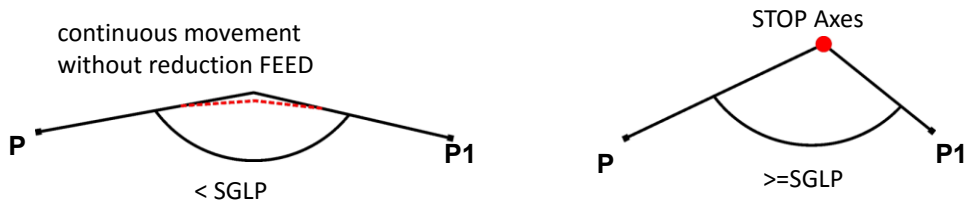
Example Parameter SGLP_RED and MAX_RED

If **SGLP_RED** is disabled (value set to ZERO) work only parameter **SGLP** or **SGL3D** in the following mode:

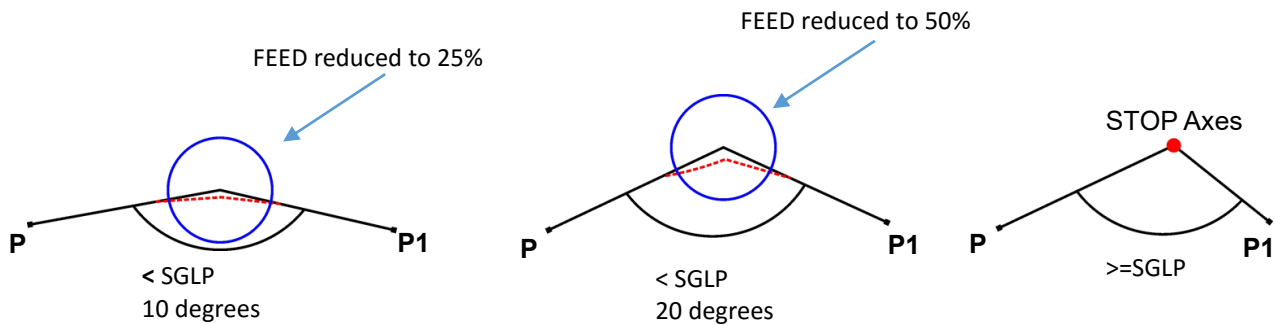
Corner less to **SGLP** or **SGL3D** the CNC not commands the stop axes

Corner greater to **SGLP** or **SGL3D** the CNC decelerates and commands the STOP Axes

Ex: **SGLP=200 (20 degrees)** , **SGLP_RED=0**



Ex: **SGLP=450 (45 degrees)** , **SGLP_RED=10 (10% of 45 degrees – at 4,5 degrees start the slowdown)**
MAX_RED=85



18.13.1.18 SGLR

Threshold to signal an error for circular interpolation.

In circular interpolation CN computer radius in start point and in end point, if the difference is greater than SGLR program is stopped showing an error. That happens when arc is made by center positions I,J with value not accurate.

The unit of measure depends by system resolution (typically 0.001 mm, default 10 thousandths). Ex:

Unit of measure 0.001 mm 10 = 10 thousandths of mm
Unit of measure 0.0001 mm 100 = 10 thousandths of mm

18.13.1.19 ACQ_MODE

It define the acquisition mode for searching of the **SENSOR** with **G102** function and for **StartAcqSens** method of Framework. Actually not in use, reserved for future expansion.

18.13.1.20 ACQ_VEL

It defines the axis speed for acquisition sensor cycle.

Contrary to other speed parameters this is NOT affected by unit of measure and must be expressed always in mm/min

18.13.1.21 RESQUOTE

Resolution for axis position. It depends by DSOF-axis parameter.

It defines the minimum unit for axis position:

10 → Tenth of mm
100 → Hundredths of mm
1000 → Thousandths of mm
10000 → 0.1 Thousandths of mm
100000 → 0.01 Thousandths of mm

18.13.1.22 VISUAREAL

Enable or disable showing of axis ACTUAL positions

-1 = Disable reading of actual position

0 = Enable by INTERFACE

1 = Enable reading of ACTUAL positions

2 = Enable reading of ACTUAL FOLLOWING ERROR position

18.13.1.23 ENABLE_OW_G0

Enable or disable override potentiometer on G0

0 = Potentiometer disabled

1 = Potentiometer enabled

2 = 2nd Potentiometer enabled

18.13.2 Axis parameters

18.13.2.1 VMAX_

It defines the maximum speed of the single axis for positioning with G0. The unit of measure depends by system resolution (typically mm/min).

This parameter is affected by unit of measure set. Working with resolution of 0.0001 (instead of typical 0.001) this parameter must be expressed in tenth of mm/min.

18.13.2.2 VMAX_Pn

It defines the maximum speed of the positioner number

18.13.2.3 VJOG_

It defines the speed of the single axis manual positioning (jogging). The unit of measure depends by system resolution (typically mm/min).

Contrary to other speed parameters this is NOT affected by unit of measure and must be expressed always in mm/min

18.13.2.4 ACC_JOG_

Acceleration for JOGGING positioning. See the above note to detail for unit of measure.

18.13.2.5 RZERO_MODE

This parameter defines the mode of homing procedure. These are the value accepted:

RZERO_MODE=0 – Backward (negative), sensor ACTIVATED

- 1) Fast backward to zero-sensor with **F=RZERO_VEL**
- 2) Slow forward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Slow backward to zero-sensor with **F=RZERO_VELF**
- 4) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 5) Preset position at **RZERO_PRESET**

RZERO_MODE=1 - Forward (positive), sensor ACTIVATED

- 1) Fast forward to zero-sensor with **F=RZERO_VEL**
- 2) Slow backward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Slow forward to zero-sensor with **F=RZERO_VELF**
- 4) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 5) Preset position at **RZERO_PRESET**

RZERO_MODE=2 - Backward (negative), sensor ACTIVATED with encoder INDEX

- 1) Fast backward to zero-sensor with **F=RZERO_VEL**
- 2) Slow forward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Slow backward to zero-sensor with **F=RZERO_VELF**
- 4) Continue to first encoder INDEX always with **F=RZERO_VELF**
- 5) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 6) Preset position at **RZERO_PRESET**

RZERO_MODE=3 - Forward (positive), sensor ACTIVATED with encoder INDEX

- 1) Fast forward to zero-sensor with **F=RZERO_VEL**
- 2) Slow backward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Slow forward to zero-sensor with **F=RZERO_VELF**
- 4) Continue to first encoder INDEX always with **F=RZERO_VELF**
- 5) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 6) Preset position at **RZERO_PRESET**

RZERO_MODE=4 - Backward (negative), sensor DISACTIVATED

- 1) Fast backward to zero-sensor with **F=RZERO_VEL**
- 2) Slow forward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 4) Preset position at **RZERO_PRESET**

RZERO_MODE=5 - Forward (positive), sensor DISACTIVATED

- 1) Fast forward to zero-sensor with **F=RZERO_VEL**
- 2) Slow backward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 4) Preset position at **RZERO_PRESET**

RZERO_MODE=6 - Backward (negative), sensor *DISACTIVATED* with encoder INDEX

- 1) Fast backward to zero-sensor with **F=RZERO_VEL**
- 2) Slow forward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Continue to first encoder INDEX always with **F=RZERO_VELF**
- 4) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 5) Preset position at **RZERO_PRESET**

RZERO_MODE=7 - Forward (positive), sensor *DISACTIVATED* with encoder INDEX

- 1) Fast forward to zero-sensor with **F=RZERO_VEL**
- 2) Slow backward to exit from zero-sensor with **F=RZERO_VELF**
- 3) Continue to first encoder INDEX always with **F=RZERO_VELF**
- 4) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 5) Preset position at **RZERO_PRESET**

RZERO_MODE=8 – Backward to encoder INDEX (no sensor)

- 1) Slow backward to first encoder INDEX with **F=RZERO_VELF**
- 2) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 3) Preset position at **RZERO_PRESET**

RZERO_MODE=9 – Forward to encoder INDEX (no sensor)

- 1) Slow forward to first encoder INDEX with **F=RZERO_VELF**
- 2) Move axis to offset position (**RZERO_OFFSET**) with **F=RZERO_VEL**
- 3) Preset position at **RZERO_PRESET**

RZERO_MODE=32 – No searching, HOME position on ENABLE DRIVE

RZERO_MODE=64 – No searching, position from ABSOLUTE ENCODER

RZERO_MODE=128+HOME_MODEx256 – CanOpen DS402 mode of HOMING procedure

18.13.2.6 LIMITE_N_

Primary Negative limit of axis position **SOFTWARE-LIMIT**. The unit of measure depends by system resolution (typically 0.001 mm, for ex. -10000 = -10mm).

18.13.2.7 LIMITE_P_

Primary Positive limit of axis position **SOFTWARE-LIMIT**. The unit of measure depends by system resolution (typically 0.001 mm, for ex. 2000000 = 2000mm).

18.13.2.8 2ND_LIMITE_N_

Secondary Negative limit of axis position **SOFTWARE-LIMIT**. The unit of measure depends by system resolution (typically 0.001 mm, for ex. -10000 = -10mm).

18.13.2.9 2ND_LIMITE_P_

Secondary Positive limit of axis position **SOFTWARE-LIMIT**. The unit of measure depends by system resolution (typically 0.001 mm, for ex. 2000000 = 2000mm).

18.13.2.10 LIMITE_N_Pn

Negative limit positioner

18.13.2.11 LIMITE_P_Pn

Positive limit of positioner

18.13.2.12 **DSOFV**

Software divider for the position of electronic manual hand-wheel. It reduces or increases its resolution. A negative value reverse the direction.

It is strictly related to VTB application, for ex:

Encoder Pulse = 100 pulse/rev.

SCALA=1 (VTB application parameter)

FILTRO=10 (VTB application parameter)

DSOFV=10000 (pulse x filtro x **10 thousandths**)

Working with default resolution of 0.001mm we obtain a hundreds for each encoder pulse.

18.13.2.13 **RZERO_OFFSET**

Value of axis positioning after homing procedure (see RZERO_MODE parameter for details).

The unit of measure depends by system resolution (typically mm).

18.13.2.14 **RZERO_PRESET**

Value of preset position after homing procedure (see RZERO_MODE parameter for details).

The unit of measure depends by system resolution (typically mm).

18.13.2.15 **RZERO_VEL**

Fast speed for homing procedure (see RZERO_MODE parameter for details).

The unit of measure depends by system resolution (typically mm/min).

18.13.2.16 **RZERO_VELF**

Slow speed for homing procedure (see RZERO_MODE parameter for details).

The unit of measure depends by system resolution (typically mm/min).

18.13.2.17 **RZERO_ACC**

Acceleration for all movement of homing procedure. See the above note to detail for unit of measure.

18.13.2.18 **MSOF_**

It defines the number of pulse of one shaft revolution. If incremental encoder are connected to CN the pulse number must be multiplied for 4 to adjust hardware decoder. It works strictly related with DSOF, indeed the ratio MSOF/DSOF is used to transform unit from pulse to millimetres.

18.13.2.19 **DSOF_**

It defines the envelope one shaft revolution. It works strictly related with DSOF, indeed the ratio MSOF/DSOF is used to transform unit from pulse to millimetres.

This value determines the unit of measure of the system. Typically it's put a value in thousandths of mm. Accordingly it must set **RESQUOTE** parameter properly.

10 → *Tenth of mm*

100 → *Hundredths of mm*

1000 → *Thousandths of mm*

10000 → *0.1 Thousandths of mm*

100000 → *0.01 Thousandths of mm*

18.13.2.20 GANTRY

It defines the index to which axis get the position value. This function can be used to enable GANTRY function. This is the relationship from index and axis name:

0 → **AXIS X**
 1 → **AXIS Y**
 2 → **AXIS Z**
 3 → **AXIS A**
 4 → **AXIS B**
 5 → **AXIS C**
 6 → **AXIS U**
 7 → **AXIS V**
 8 → **AXIS W**

To enable GANTRY function:

GANTRY_X → 0=disabled or index of axis master
GANTRY_Y → 1=disabled or index of axis master
GANTRY_Z → 2=disabled or index of axis master
GANTRY_A → 3=disabled or index of axis master
GANTRY_B → 4=disabled or index of axis master
GANTRY_C → 5=disabled or index of axis master
GANTRY_U → 6=disabled or index of axis master
GANTRY_V → 7=disabled or index of axis master
GANTRY_W → 8=disabled or index of axis master

ATTENTION!!!

To enable GANTRY function it must be supported by VTB application on CN.

18.13.2.21 SGL3D

It defines the threshold to detection of **EDGE** in 3D interpolations. This is used by **G65 G67** or by **AFC** filter **G66**. The value is approximatively expressed in tenth of degree. The proper value must be found from dynamics of the machine. In **G65** or **G67** interpolation when an axis overcomes this threshold axis are stopped.

Reference value for SGL3D

THRESHOLD in DEGREES	VALUE OF SGL3D (min-max)
5	60-90
10	125-175
20	250-350
30	300-500
45	400-700

18.13.2.22 SGLAFC

It defines the threshold for **AFC G66** to automatic reduction of the speed. This value determines the maximum speed change of each axis tolerates. When in the path are found values higher than SGLAFC speed is reduced. The value depends by dynamic of machine.

18.13.2.23 MILD

It Defines the MILD MODE value ([see G49 functions](#))

Estimated values from 200 to 20

0 or 1000 DISABLED

18.13.2.24

BACKLASH

Parameter for BACKLASH compensation. The unit of measure depends by system resolution (typically 0.001 mm, for ex. -10 = 0.01mm).

18.13.2.25

TBCK

It define a time to achieve the BACKLASH recovery. If $TBCK \geq BACKLASH$ it is recovered in a single CN sample, else it is divided in more samples. The number of

samples will be: **BACKLASH / TBCK**

With the result rounded-up.

Ex:

BACKLASH=100

TBCK=80

N.SAMPLES=1.25 → 2 samples

Sample is defined in the VTB application on CN (typically 2 msec).

It's recommended to distribute the BACKLASH recovery in more sample when it is high (over 3-4 hundredths).

18.13.2.26

TSHF

It define a time to achieve the SetShiftAxis recovery.

This value indicate the increment xTAU

Ex:

TAU=2 Ms

ValShift=100

TSHF_=10

The entire position is reached in 20 Ms

18.13.2.27

WR_SPD9

It define automatic write SPEED "S" to variable USER GENERIC 10 to NC.

WR_SPD=0 Automatic write disabled

WR_SPD=1 Automatic write Enabled

If WR_SPD=1 when the S value of PartProgram is automatically writing to NC Variable User Generic 10

18.13.2.28

RFG

It define automatic limitation FEED in G1-G2-G3

RFG=0 Whitout limitation

RFG=1 If the velocity exceeds that of the single axis (VMAX_), it is limited

18.13.2.29

JERK

It define the value of JERK for S Ramp.

If the value is ZERO the S ramp is disabled (use trapezoidal ramp)

Value > 0 enable S ramp.

Max value 100

18.13.2.30

CR_LIMIT

Enables or disables the prior control of axis limits

If enabled, each START, PART PROGRAM is not performed if the axes exceed the limits set machine

CR_LIMIT=0 Disable

CR_LIMIT=1 Preventive control G0-G1 activated only on the final point of G2-G3

CR_LIMIT=2 Preventive control activated G0-G1-G2-G3 (with explosion of ARCS)

CR_LIMIT=3 Preventive control activated G0-G1-G2-G3 only in run time, before executed the block (G2-G3 Only final point is controlled)

18.13.2.31 ARC_REL

Enables or disables the center of arc I,J in relative mode

ARC_REL=0 The center I,J is in absolute mode (if set G90)

ARC_REL=1 The center I,J is in relative mode (if set G90)

18.13.2.32 NO_SHORT

Enables or disables the CNC to remove the short segments

NO_SHORT=0 Function disactivated

NO_SHORT=1 The short segment are removed by CNC. This facilitates the machining of complex contours

NO_SHORT=2 Only Warning message is displayed if the short segment is worked

18.13.2.33 USE_G60

If the value is >0, enabled default mode G60 and not G61

18.13.2.34 TIME_OUT_CMD

TimeOut Commands invoked on CNC. Value in Ms (default 5000)

18.13.2.35 TIME_OUT_M

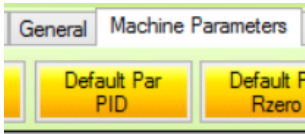
TimeOut **M** invoked on CNC.. Value in Ms (default 5000)

18.14 PID PARAMETERS

The PID Parameters are used for the analogical Axes velocity control with Encoder Loop.

Usually, these Axes are controlled by +/- 10 Analog Outputs.

These Parameters are inserted by IsoNs configurator “Machine Parameters→Default Par PID”:



18.14.1.1 PID_KP

Proportional Gain

18.14.1.2 PID_KI

Integrative Gain

18.14.1.3 PID_KV

Feed Forward Gain.

For a correct setting You have to use the following method:

- 1) Set to ZERO (0) the PID_KP and PID_KI
- 2) Insert a PID_KV value
- 3) Move the axes and check if the real position is near to target (+/- 5 %)
- 4) Insert the PID_KP and PID_KI

WARNING

This operation must be did with Axis disconnected from mechanical part

18.14.1.4 PID_I_LIMIT

Integrative Limit

18.14.1.5 PID_DIV

PID parameters Magnitude. Increase this parameter for Increase the PID parameter resolution:

Ex: PID DIV=10 PID KP=1

It is the same

PID DIV=100 PID KP=10

18.14.1.6 PID_SERVO

Maximum following error (micron).

If the Axis exceeds this value, an alarm is invoked

(must be exceeds this value for a PID TIME SERVO time)

18.14.1.7 PID_TIME_SERVO

If the Axis exceeds the PID_SERVO for a time lower of PID_TIME_SERVO (Millisecond)

The ERROR isn't invoked

18.14.1.8 PID DIR

Analog Outputs Direction (0 or 1)

Set this parameter for tuning the Axis direction with Analog Output

WARNING

For change the Axis direction (ex: CW in CCW) is necessary Change this Parameter and also change the SIGN of MSOF parameter (ex: 5000 to -5000)

18.14.1.9 PID_OFFS_ANA

Reset the analog output Offset.

With all PID parameters (KP,KI) set to ZERO, set this parameter until the axis is stationary (when it is enabled)

Index

2.1	BLOCK.....	3
2.2	LINE NUMBER OR BLOCK NUMBER.....	3
2.3	PROGRAM RUNNING.....	3
2.4	AXIS NAME.....	3
2.5	HOMING.....	3
2.6	WORK ORIGIN.....	3
2.7	WORK OFFSET.....	3
2.8	HEADS.....	3
2.9	MODAL FUNCTIONS.....	4
2.10	RECOGNIZED CODES.....	4
2.11	NUMERIC VALUES SETTING.....	4
2.12	PROGRAM REMARKS.....	4
3.1	RECOGNIZED G CODES.....	5
4	OTHERS RECOGNIZED CODES.....	7
4.1	PROGRAM FLOW INSTRUCTIONS.....	10
4.2	GENERIC INSTRUCTIONS.....	11
4.3	MATH AND LOGICAL OPERATORS.....	12
4.4	MATH FUNCTIONS.....	13
4.5	VARIABLES AND CONSTANTS.....	14
4.6	PREDEFINED VARIABLES.....	14
4.7	NSFORMS INSTRUCTIONS.....	15
5.1	IF-ELSE-END_IF.....	16
5.2	LOOP - END_LOOP.....	16
5.3	GOTO.....	17
5.4	GOSUB – RETURN.....	17
5.5	LABEL.....	18
5.6	END_PROGRAM.....	18
5.7	WAIT_INPUT.....	18
5.8	ERROR.....	19
5.9	RESUME_T.....	19
5.10	SAVE_T.....	19
6.1	SDO_DL.....	20
6.2	SDO_UL.....	20
6.3	GET.....	20
6.4	READ_PARMAC.....	21

6.5	WRITE_PARMAC	21
6.6	OPT.....	21
6.7	PAUSE_MODE	22
6.8	IMPORT	22
6.9	END_IMPORT	22
7.1	LIB.MESSAGE	23
7.2	LIB.SHOWFORM	24
7.3	LIB.CLOSEFORM	24
7.4	LIB.FORMPROP	24
7.5	LIB.FORMTEXT	25
7.6	LIB.ADDLABEL.....	25
7.7	LIB.LABELPROP	25
7.8	LIB.LABELTEXT	26
7.9	LIB.LABELPRINT	27
7.10	LIB.ADDBUTTON	27
7.11	LIB.BUTTONPROP.....	27
7.12	LIB.BUTTONTEXT.....	29
7.13	LIB.BUTTONPRINT	29
7.14	LIB.ADDINPUT.....	29
7.15	LIB.INPUTPROP	30
7.16	LIB.INPUTSETVALUE	31
7.17	COLOR TABLE	32
8.1	SIN.....	34
8.2	COS.....	34
8.3	LOG	34
8.4	EXP	34
8.5	INT.....	34
8.6	FIX	34
8.7	ABS.....	35
8.8	DRG	35
8.9	RAD	35
8.10	SQR.....	35
8.11	TAN	35
8.12	ATAN	35
8.13	ASIN.....	35
8.14	ACOS.....	36
9.1	ROUND BRACKETS ()	37

9.2	ISO EXPRESSIONS AND SQUARE BRACKETS []	37
9.3	VARIABLE TEST	37
9.4	BIT CONTROL	38
10.1	NUMERIC CONSTANTS	38
10.2	GENERIC VARIABLES "\$"	38
10.3	AXIS POSITION VARIABLES	39
10.4	DIGITAL INPUT/OUTPUT VARIABLES	39
10.5	TIMER VARIABLES	40
10.6	TOOL TABLE VARIABLES	41
10.7	HEAD TABLE VARIABLES	42
10.8	AXIS COUNTER VARIABLES	42
11	LIMIT AXIS VARIABLES	43
11.1	GLOBAL VARIABLE	43
11.2	M FUNCTION VARIABLES (FOR CN)	43
11.3	SPECIAL PARAMETERS VARIABLES	43
11.4	WORK ORIGIN AND OFFSET VARIABLES	45
11.5	USER GENERIC VARIABLES	45
11.6	ARRAY VARIABLES - DIM	45
11.7	MARKER VARIABLES	45
13	PA-PD(N,PAR)ESPR SET ABSOLUTE POSITION	47
14	PF(N)ESPR SET POSITIONER FEED	47
15	PS(N) STOP POSITIONER	47
16	_PM(N,PAR) READ STATUS POSITIONER	47
17.1	DEFINITION OF AXIS POSITIONS	48
17.1.1	<i>G90 – PROGRAMMING WITH ABSOLUTE POSITIONS</i>	48
17.1.2	<i>G91 - PROGRAMMING WITH INCREMENTAL POSITIONS</i>	48
17.1.3	<i>Definition of absolute and incremental positions</i>	49
17.2	WORK ORIGIN	49
17.2.1	<i>Work origin By INDEX</i>	49
17.2.2	<i>G94 – Work origin at position defined with parameter</i>	50
17.2.3	<i>G54, G55, G56, G57, G58, G59 - Work origin from memory file</i>	51
17.2.4	<i>G92 – Work origin in current axis position</i>	52
17.2.5	<i>G82 – Work origin in current axis position with sensor offset</i>	52
17.2.6	<i>G98 – G53 - Suspend work origin</i>	52
17.2.7	<i>G99 – Restore work origin</i>	52
17.2.8	<i>G940 MOVE axes excluding WORK ORIGIN only in the actual block</i>	53
17.2.9	<i>USER_ZERO – Index of WORK ORIGIN LIST</i>	53
17.3	WORK OFFSET	54
17.3.1	<i>G93 - Work offset at position defined with parameter</i>	54
17.3.2	<i>G95 - Work offset in current axis position</i>	54
17.3.3	<i>G85 - Work offset in current axis position with sensor offset</i>	54

17.3.4	<i>G86 - Hardware preset axis on module 360 degrees</i>	55
17.3.5	<i>G96 - Suspend work offset</i>	55
17.3.6	<i>G97 - Restore work origin</i>	55
17.3.7	<i>USER_OFFSET - Index of WORK OFFSET LIST</i>	55
17.4	WORK HEAD SELECTION – H FUNCTION	57
17.4.1	<i>Hn – Select head</i>	57
17.4.2	<i>G87 – Suspend head offset</i>	57
17.4.3	<i>G88 – Restore head offset</i>	57
17.5	ROTATIVE AXIS	58
17.5.1	<i>G36 – Rotative axis definition</i>	58
17.6	HARDWARE PRESET OF AXIS	59
17.6.1	<i>G89 – Hardware preset of axis position</i>	59
17.7	HARDWARE PRESET OF AXIS	59
17.7.1	<i>G84 – Preset Axes counters</i>	59
17.7.2	<i>G83 – Preset CPU 1 counters</i>	59
17.8	SELECT WORK PLANE	60
17.8.1	<i>G17 – Select work plane on X-Y</i>	60
17.8.2	<i>G18 - Select work plane on X-Z</i>	60
17.8.3	<i>G19 - Select work plane on Y-Z</i>	60
17.8.4	<i>G70 - Select work plane on axis by parameters</i>	60
17.9	START SEARCH OF HOME POSITION	61
17.9.1	<i>G71 – Start Homing cycle</i>	61
17.10	AXIS MOVEMENT FUNCTIONS	62
17.10.1	<i>G0 – Rapid positioning</i>	62
17.10.2	<i>G1 – Linear interpolation a programmed F speed</i>	62
17.10.3	<i>G2/G3 - Circular interpolation a programmed F speed</i>	63
17.10.4	<i>G30 - Enable automatic insert of fillet on edges</i>	64
G31 – SUSPEND OF AUTOMATIC INSERT OF FILLET		65
17.10.5	<i>G32 - Restore of automatic insert of fillet</i>	65
17.10.6	<i>G33 - Enable automatic insert of bevel on edges</i>	66
17.10.7	<i>G34 - Suspend of automatic insert of bevel</i>	67
17.10.8	<i>G35- Restore of automatic insert of bevel</i>	67
17.10.9	<i>G102 – Start searching sensor position</i>	68
17.11	PROGRAMMING AXIS INTERPOLATION SPEED	69
17.11.1	<i>F – Speed of axis interpolation</i>	69
17.11.2	<i>ARC speed auto-reduction</i>	69
17.12	WORK PLANE TRANSFORMATION	70
17.12.1	<i>G120 – Vertical mirror</i>	70
17.12.2	<i>G121 – Disable vertical mirror</i>	70
17.12.3	<i>G24 – Horizontal mirror</i>	71
17.12.4	<i>G25 – Disable horizontal mirror</i>	71
17.12.5	<i>G22 – Echange axis of work plane</i>	71
17.12.6	<i>G23 – Restore axis of work plane</i>	72
17.12.7	<i>G26 – Exchange axis by parameter</i>	72
17.12.8	<i>G27 – Suspend G26</i>	72
17.12.9	<i>G28 – Restore G26</i>	72

17.12.10	<i>G51 – Suspend work plane rotation</i>	73
17.12.11	<i>G52 – Restore work plane rotation</i>	73
17.12.12	<i>G50 - Work plane rotation</i>	74
17.12.13	<i>G1050 – Disable Scaling</i>	76
17.12.14	<i>G1051 – Enable Scaling</i>	76
17.12.15	<i>G103 – Set RTCP parameters</i>	78
17.12.16	<i>G104 – Enable/restore RTCP</i>	78
17.12.17	<i>G105 – Disable RTCP</i>	78
18	TOOL RADIUS COMPENSATION	79
18.1.1	<i>G41/G42 – Enable left/right radius tool compensation</i>	79
18.1.2	<i>G40 – Disable radius tool compensation</i>	79
18.1.3	<i>D – Tool diameter</i>	80
18.1.4	<i>G47 – Set tool entry mode</i>	81
18.2	TOOL LENGHT COMPENSATION	82
18.2.1	<i>G43 – Enable tool length compensation from parameter</i>	82
18.2.2	<i>G44 – Disable tool length compensation G43</i>	83
18.2.3	<i>G45 – Enable tool length compensation from tool table T</i>	83
18.2.4	<i>G46 – – Disable tool length compensation G45</i>	84
18.3	INTERPOLATION MODES	85
18.3.1	<i>G60 – Enable FAST interpolation without stop on segments</i>	85
18.3.2	<i>G61 – Enable interpolation with stop on segments</i>	85
18.3.3	<i>G62 – Wait for stop axis</i>	85
18.3.4	<i>G63 – Interpolation outside work plane (PX_MOVETO)</i>	86
18.3.5	<i>G64 – Interpolation on work plane (default)</i>	86
18.3.6	<i>G65 – Enable 3D interpolation PX_MOVETO with calculated stop on edge by parameters</i>	87
18.3.7	<i>G75 – Enable G64 for movement inside the work plane , G65 for movement outside the work plane</i>	87
18.3.8	<i>G66 – AFC – Adaptive Feed Control</i>	88
18.3.9	<i>G67 – Px_Moveto for movement outside plane and Px_Lineto for inside one</i>	89
18.3.10	<i>G68 – Always using of PX_LINETO in G1 “TRANSPORTED AXIS” (with possibility to combine with PX_MOVETO</i>	89
18.3.11	<i>G69 – LHK – Buffer look ahead depth</i>	90
18.4	ISONS FILTERS	91
18.4.1	<i>G72 – N.U.R.B.S (Non Uniform Rational Bspline) (2D 3D)</i>	92
18.4.2	<i>G73 – NOISE (2D)</i>	95
18.4.3	<i>G74 – RLS Remove Len Segment (2D 3D)</i>	97
18.4.4	<i>G106 – Smoothing (2D 3D)</i>	98
18.5	VIRTUAL AXIS	101
18.5.1	<i>G100 – Comando sincrono per asse virtuale</i>	101
18.6	OTHER GENERIC G FUNCTIONS	102
18.6.1	<i>G4 – Timed pause</i>	102
18.6.2	<i>G10 – Enable external OVERRIDE potentiometer</i>	102
18.6.3	<i>G11 – Disable external OVERRIDE potentiometer</i>	102
18.6.4	<i>G101 – Axis Stop command</i>	103
18.6.5	<i>G80 – Forced pause by code</i>	103
18.6.6	<i>G81 – Management secondary axes LIMITS</i>	104
18.6.7	<i>G20 – Axes Values in Inch</i>	104
18.6.8	<i>G21 – Axes Values in Millimeters</i>	104
18.7	VARIABLE MANAGEMENT FUNCTIONS	105

18.7.1	LOAD_VAR – load a variables file in current list	105
18.7.2	GET_VAR – Read a value from the loaded list and store it in a variable	105
18.7.3	WRITE_VAR – Write a value in the loaded list getting it from a variable	106
18.7.4	SAVE_VAR – save a variables file with current list	106
18.7.5	FILE_EXISTS – test if a file exists	106
18.7.6	ADD_VAR – Add a value to current list	107
18.7.7	REMOVE_VAR – Remove a value from current list	107
18.7.8	CLEAR_VAR – Remove all values from the current list	107
18.7.9	DIM_VAR – Size current list to a specific number of elements	108
18.7.10	COUNT_VAR – Get the number of elements in the current list	108
18.8	HM FUNCTIONS	109
18.8.1	Call of HM functions	109
18.8.2	Building of a HM function	110
18.9	M FUNCTIONS	111
18.9.1	M functions inside CN	111
18.9.2	M function inside PC	112
18.10	ESSENTIAL M FUNCTIONS	113
18.10.1	START M	113
18.10.2	END M	113
18.10.3	STOP M	113
18.10.4	PAUSE M	113
18.10.5	RESUME PAUSE M	114
18.10.6	GO BLOCK M	114
18.10.7	GO RETRACE M	114
18.11	DEFINING DEPTH AXIS	115
18.11.1	G48 Define Depth axis	115
18.12	MILD MODE – EDGE SMOOTHING	116
18.12.1	G49 MILD MODE Managing	116
18.13	MAIN MACHINE PARAMETERS	118
18.13.1	Generic parameters	118
18.13.2	Axis parameters	121
18.14	PID PARAMETERS	128